NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

Approved for public release; distribution is unlimited.

20010627 072

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE March 2001	3. REPORT TYPE AND DATES COVERED Masters Thesis	
4. TITLE AND SUBTITLE Modeling Tactical Land Combat Using A Multi-Agent System Design Paradigm		5. FUNDING NUMBERS	
6. AUTHOR(S) Pawloski, Joel S.			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000		8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)		10. SPONSORING / MONITORING AGENCY REPORT NUMBER	

11. SUPPLEMENTARY NOTES
The views expressed in this thesis are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.	12b. DISTRIBUTION CODE

13. ABSTRACT *(maximum 200 words)*

In the past 60 years the Army has undergone a major reorganization eight times at the divisional level and numerous more times at unit levels below the division. Each time the Army reorganized it's divisions a major testing program was involved. But when a change in organization is done at unit levels below division often very little attention is paid to how the change will affect the unit. When this happens, unit leaders are forced to undertake one of the most difficult jobs in today's military incorporating new equipment into a unit or reorganizing a unit without an understanding of how the changes will affect the unit.

The Military modeling and simulation community has attempted to fill this need but the current set of single entity simulations are limited in their ability to replicate dynamic complex behavior. This thesis is attempting to create a Multi-Agent Simulation that will allow analysts and leaders to gain an understanding of the tactical employment affects of changing the organization of a company level infantry unit.

GIAgent is a simulation tool allowing the analyst and leader to experiment with the complex relationship between maneuver and unit organization without putting the unit in the field. Combining agent based artificial intelligence techniques with artificial intelligence research from the computer gaming industry, GI Agent creates a new paradigm for combat simulation.

The GIAgent software uses the *RELATE* architecture designed by LCDR Kim Roddy, USN and Lt Mike Dixon, USN.

14. SUBJECT TERMS Multi-agent system, MAS, Combat Modeling, Human and Organizational Behavior, Agent-Based Simulation, Adaptive Agents, Autonomous Agents.			15. NUMBER OF PAGES 104
			16. PRICE CODE
17. SECURITY CLASSIFICA-TION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

MODELING TACTICAL LAND COMBAT USING A MULTI-AGENT SYSTEM DESIGN PARADIGM (GI AGENT)

Joel S. Pawloski
Captain, United States Army
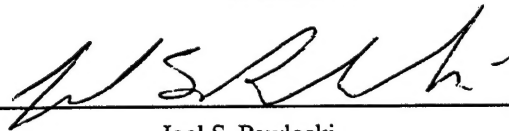B.S., Embry-Riddle Aeronautical University, 1990

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN
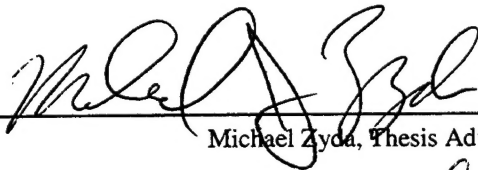## MODELING, VIRTUAL ENVIRONMENTS AND SIMULATION

from the

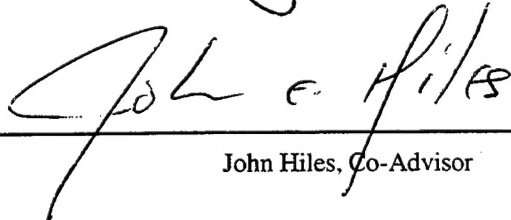## NAVAL POSTGRADUATE SCHOOL
March 2001

Author: _____
Joel S. Pawloski

Approved by: _____
Michael Zyda, Thesis Advisor

_____
John Hiles, Co-Advisor

_____
Rudy Darken, Academic Associate
Modeling, Virtual Environments, and Simulation Academic Group

_____
Michael Zyda, Chairman
Modeling, Virtual Environments and Simulation Academic Group

iii

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

In the past 60 years the Army has undergone a major reorganization eight times at the divisional level and many more times at unit levels below the division. Each time the Army reorganized it's divisions a major testing program was involved. But when a change in organization is done at unit levels below division often very little attention is paid to how the change will affect the unit. When this happens, unit leaders are forced to undertake one of the most difficult jobs in today's military incorporating new equipment into a unit or reorganizing a unit without an understanding of how the changes will affect the unit.

The military modeling and simulation community has attempted to fill this need but the current set of single entity simulations are limited in their ability to replicate dynamic complex behavior. This thesis is attempts to create a Multi-Agent Simulation that allows analysts and leaders to gain an understanding of the tactical employment effects of changing the organization of a company level infantry unit.

GIAgent is a simulation tool allowing the analyst and leader to experiment with the complex relationship between maneuver and unit organization without putting the unit in the field. Combining agent based artificial intelligence techniques with artificial intelligence research from the computer gaming industry, GI Agent creates a new paradigm for combat simulation.

The GIAgent software uses the *RELATE* architecture designed by LCDR Kim Roddy, USN and Lt Mike Dixon, USN.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

LIST OF DEFINITIONS

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ABBREVIATIONS AND ACRONYM'S

AFS	Abstract Force Simulator
AI	Artificial Intelligence
CAS	Complex adaptive systems
DMSO	Defense Modeling and Simulation Office
DoD	Department of Defense
GA	Genetic Algorithm
GUI	Graphic User Interface
HAC	Hierarchical Agent Control
ISAAC	Irreducible Semi-Autonomous Adaptive Combat
KIA	Killed in Action
LOS	Line of Sight
M&S	Modeling and Simulation
MAS	Multi-Agent System
MOE	Measure of Effectiveness
MOVES	Modeling, Virtual Environments and Simulation
NPS	Naval Postgraduate School
SFI	Santa Fe Institute
USN	United States Navy
VE	Virtual Environment
WIA	Wounded in Action

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. PROBLEM STATEMENT

The purpose of this thesis is to create the basis for modeling tactical entities in a Multi-Agent Simulation. For a given a combination of infantry personnel, equipment, and tactical situation, this simulation provides insights into the organizational structure of an infantry company. The purpose of this thesis is to assist in making the most effective unit possible.

B. MOTIVATION

In the past 60 years the Army has undergone a major reorganization eight times at the divisional level and many more times at unit levels below the division. Each time the Army reorganized it's division a major testing program was involved. But when a change in organization is done at unit levels below division often very little attention is paid to how the change will affect the unit. Often unit leaders are forced to undertake one of the most difficult jobs in today's military incorporating new equipment into a unit or reorganizing a unit without an understanding of how the changes will affect the unit.

Three times in my career I have been in a unit that has undergone a significant organizational or equipment change. In each instance, it was unclear how the changes would affect the tactical employment of the unit. The leaders of the units were forced to experiment with tactical employment "on the job" at the National Training Center or possibly in combat. Fighting a company is a very demanding job even under the best

1

conditions. If a commander is unsure of the best possible way to fight a unit, the mistakes that are made will cost lives.

Traditional modeling techniques have been unable to represent the complex adaptive behavior possible with a multi-agent system. Based on the work of Ilachinski and others it has been shown that land combat can be represented by using Multi-Agent System. The justification for creating a new multi-agent system is to design one with dynamic organizational structure. Traditional techniques and ISAAC are not capable of dynamic organizational structure.

In this thesis, a multi-agent simulation was created that allows analysts and leaders to gain an understanding the tactical employment affects of changing the organization of a company level infantry unit. Thus allowing the analyst and leader to experiment with the organization without putting the unit in the field and possibly make better use of the time spent in the field.

C.   GOALS

The main goals of this thesis are summarized as follows:

- Develop a Multi-Agent Simulation capable of depicting a realistic deployment of an infantry company.

- Develop a JAVA based library of classes that can be the basis of a MAS toolkit for simulating entity level tactical combat.

- Demonstrate the applicability of this thesis by conducting an organizational experiment of a simulated infantry company.

2

D. ORGANIZATION

Chapter II is a review of background material and similar work supporting this thesis. Chapter III outlines the architecture of the GI Agent MAS and discusses the major algorithms used in the simulation. Chapter IV describes the GI Agent Organizational Experiment and discusses the insights gained from the organizational experiment. Chapter V provides the conclusions, lessons learned and the recommended areas for future research.

THIS PAGE INTENTIONALLY LEFT BLANK

II. BACKGROUND

A. INTRODUCTION

In order to discuss the relevance of GI Agent, it is first necessary to put it into practical perspective. My own experience as a Scout Platoon Leader and Air Cavalry Troop Commander provide a basis for the creation of the GI Agent Multi-Agent System simulation. As a Scout Platoon Leader during a National Training Center rotation, I was given command of an electronic warfare section, chemical reconnaissance section, mortar section, and combat engineer platoon in addition to my scout platoon. Instantly, my platoon ballooned from 27 men to 72 men and from 6 vehicles to 14 vehicles. The result was an organizational nightmare and tactically unwieldable unit as different sections attached to my platoon had different and incompatible missions.

During my time as commander of an Air Cavalry Troop, the unit underwent a dramatic change in organization. The air scout platoon and the attack platoon were replaced by two platoons of OH-58D Kiowa Warriors. These new aircraft resulted major combat employment changes for the troop. As the company commander I was required to learn how to employ the troop by trail and error. This resulted in a significant portion of the troop training time being wasted.

Another example of organizational instability at the tactical level was in the Russian assault on Gronzy in January of 1995. The initial assault on New Year's Eve had failed to take the city. So the Russians regrouped, consulted their doctrine and attacked again. This time following untested doctrine for urban combat, which called for the formation of a new combat task force called a *storm group.* A *storm group* is a

motorized rifle company reinforced with a tank platoon, artillery battery, mortar platoon, automatic grenade launch platoon, engineer platoon, and chemical troops. Combat experience with these formations showed that the creation of the *storm groups* was counterproductive. Unit integrity was destroyed and company commanders were saddled with more assets than they could effectively manage. After several unsuccessful assaults in which many lives were lost, the Russian command determined that a better solution was to use the base motorized rifle company and reinforce the company with specialty troops as needed based on the current mission.

Although the *storm groups* looked great on paper, the dynamic environment of combat proved that the organization was too complex to be managed by a company commander who lacked experience in the operation of such a unit. The company commander in charge of a *storm group* was responsible for the tactical employment of units that he had never worked with prior to the formation of the *storm group*. If these *storm groups* had been tested in a dynamic MAS combat simulation prior to combat conditions the weaknesses of the organization could have been identified and lives saved.

B. KEY IDEAS AND DEFINITIONS

The definitions of several key concepts are required to be understood by the reader prior to reading this thesis. These concepts are the basis for the design of this thesis. Agent, situated agent and multi-agent system are basic building blocks of the thesis. Organization of a tactical level unit and the organizational analysis of that unit are the function of this thesis.

1. Agent

The primary building block of any Multi-Agent System or *complex adaptive system* is the adaptive agent. Agents are software constructs designed to operate semi-independently. A key element of agent design is the ability of agents to interact in a cooperative or competitive fashion. There are however many definitions and many different types of agents I will constrain this discussion to only the relevant type and a single definition.

The following is the definition for agent (Ferber 1999) that I will use:

An agent is a physical or virtual entity
  (a) which is capable of acting in an environment,
  (b) which can communicate directly with other agents,
  (c) which is driven by a set of tendencies (in the form of individual objectives or of satisfaction/survival function which it tries to optimize),
  (d) which possesses resources of its own
  (e) which is capable of perceiving its environment (but to a limited extent),

  (f) which has only a partial representation of this environment (and perhaps none at all)
  (g) which possesses skills and can offer services,
  (h) which may be able to reproduce itself,
  (i) whose behavior tends towards satisfying its objectives, taking account of the resources and skills available to it and depending on its perception, its representations and the communications it

The agents in this thesis are artificial soldiers. Each agent is given a set of goals and corresponding rules and the resources to attempt to achieve those goals.

2. Situated Agent

The agents in GI Agent are artificial soldiers situated in a notional terrain based environment. Each agent works in a cooperative fashion with the other agents in the same notional organization. Situated agents are a subclass of agent.

7

Situated agents are defined by Ferber (1999) as follows:

A purely situated agent is defined as a physical entity (or perhaps a computing entity if it is simulated) which
- (a) is situated in an environment
- (b) is driven by a survival/satisfaction function,
- (c) possesses resources of its own, in the form of power and tools,
- (d) is capable of perceiving its environment (but to a limited extent),
- (e) has practically no representation of its environment,
- (f) possesses skills,
- (g) can perhaps reproduce,
- (h) has behavior tending to fulfill its survival/satisfaction function, taking into account the resources, perceptions and skills available to it.

3. Multi-Agent System

The core of this thesis is a Multi-agent System (MAS) that is based on and loosely replicates a dismounted infantry company operating on different types of terrain. Ferber defines the MAS in terms of the elements environment (E), objects (O), agents (A), relations (R), operations (Op), and Laws.

The formal definition (Ferber 1999) is as follows:

The term "multi-agent system (or MAS) is applied to a system comprising the following elements:
- (1) An environment, E that is a space which generally has a volume.
- (2) A set of objects, O. these objects are situated, that is to say, it is possible at a given moment to associate any object with a position in E. These objects are passive that is they can be perceived, created, destroyed and modified by the agents.
- (3) An assembly of agents, A, which are specific objects representing the active entities of the system.
- (4) An assembly of relations, R, which link objects (and thus agents) to each other.
- (5) An assembly of operations, Op, making it possible for the agents of A to perceive, produce, consume, transform, and manipulate objects from O.
- (6) Operators with the task of representing the application of these operations and the reaction of the world to this attempt at modification, which we shall call the laws of the universe.

8

The environment in this thesis environment consists of a specified area of terrain. Objects in the MAS are buildings, objective markers and the troops. The agents are the individual soldiers. Relations in the MAS are defined by the chain of command relationships that exist in a real infantry company. The agents operate in the environment by moving, shooting, and communicating. A variety of laws are in place in the MAS such as line-of-sight.

4.   MAS Simulation

Multi-Agent Systems are often used to model complex environments or phenomenon in a way that traditional computer modeling is incapable of doing. Often, an agent-based model is used to investigate the environment or situation at the micro level. With traditional mathematical modeling, large numbers of parameters can cause instability in the model and as a result, infeasibility in attempting to model the given phenomenon. MAS simulations generally use a simpler mathematical approach and achieve complexity through agent interactions.

The main reasons for using a Multi-Agent system as a modeling environment are its capacity for integration and the flexibility of the technique. However the most unique aspect of MAS simulation is the possibility of creating a model of macro action based entirely on the micro interactions of modeled entities.

The following definition of a MAS simulation will be used (Hiles, 1999):

MAS Simulation: A rich, bottom-up modeling technique that uses diverse, multiple agents to imitate selected aspects of the real world system's active components.

9

5.   Organization

An agent organization could be any collection of agents. This definition is to loose for the purposes of this thesis. Agent organizations are usually defined by the roles the agents play in the organization and the relationships between the agents in a multi-agent system. The key element to an agent organization is the interrelationships that exist between the agents. These interrelationships prescribe how an agent organization will react to changes in the environment. Organization describes the process of building a structure and the result of the process of building. Organization is by necessity a dynamic entity and is capable of reorganization in response to stimuli.

Webster's dictionary defines organization as an administrative and functional structure or as the personnel of such a structure. This definition is insufficient for the purposes of the thesis so a more specific definition will be used.

> **Organization:** An organization can be defined as an arrangement of relationships between components or individuals which produces a unit, or system, endowed with qualities not apprehended at the level of the components or individuals. The organization links, in an interrelational manner, diverse elements or even events or individuals, which henceforth become the components of a whole. It ensures a relatively high degree of interdependence and reliability, thus providing the system with the possibility of lasting for a certain length of time, despite chance disruptions (Morin, 1977).

6.   Organizational Analysis

The functional analysis of an organization can be done on several levels, primarily internal functions and external function. Analysis of internal functions could be

10

defined as the study of the interactions between agents in an organization or in analyzing the efficiency of actions in an organization.

This thesis studies the external functions of an organization. This thesis analyzes the efficiency of the actions of an organization as a whole interacting with the environment.

C.   SIMILAR MULTI-AGENT SYSTEMS

1.   ISAAC

> The central thesis of this report is that land combat can be thought of as a complex adaptive system. – Military conflicts, particular land combat, have all of the key features of complex adaptive systems: combat forces are composed of large numbers of nonlinearly interacting parts and are organized in a command and control hierarchy; local action, which often appears disordered, induces long-range order (i.e. combat is self-organized) military conflicts, by their nature, proceed far from equilibrium; military forces, in order to survive, must continually adapt to a changing combat environment; there is no master "voice" that dictates that actions of each and every combatant.   - ANDREW ILACHINSKI

As the above quote states ISAAC (Irreducible Semi-Autonomous Adaptive Combat) is one of the first attempts to model land combat as a Multi-agent System. The primary goal of ISAAC is to gain an understanding of the fundamental processes of modern land warfare using a bottom-up synthesis approach. The question Ilachinski is attempting to answer with ISAAC is *"To what extent is land combat a self-organized emergent phenomenon?"* (Ilachinski, 1997).

ISAAC represents an initial effort toward developing a "complex systems theoretic analyst's toolbox (or "conceptual playground") for exploring high-level

11

emergent collective patterns of behaviors arising from various low-level (i.e., individual combatant and squad-level) interaction rules."

ISAAC represents combat as abstract blue or red entities (ISAACA) fighting on a two dimensional battlefield. Each agent evaluates the space around it and based on its internal goals reacts to the environment and changes in the environment. The agents are limited in what they can "see" and thus only can respond to changes in the local environment.

Below is a representation of an ISSACA and its local environment.
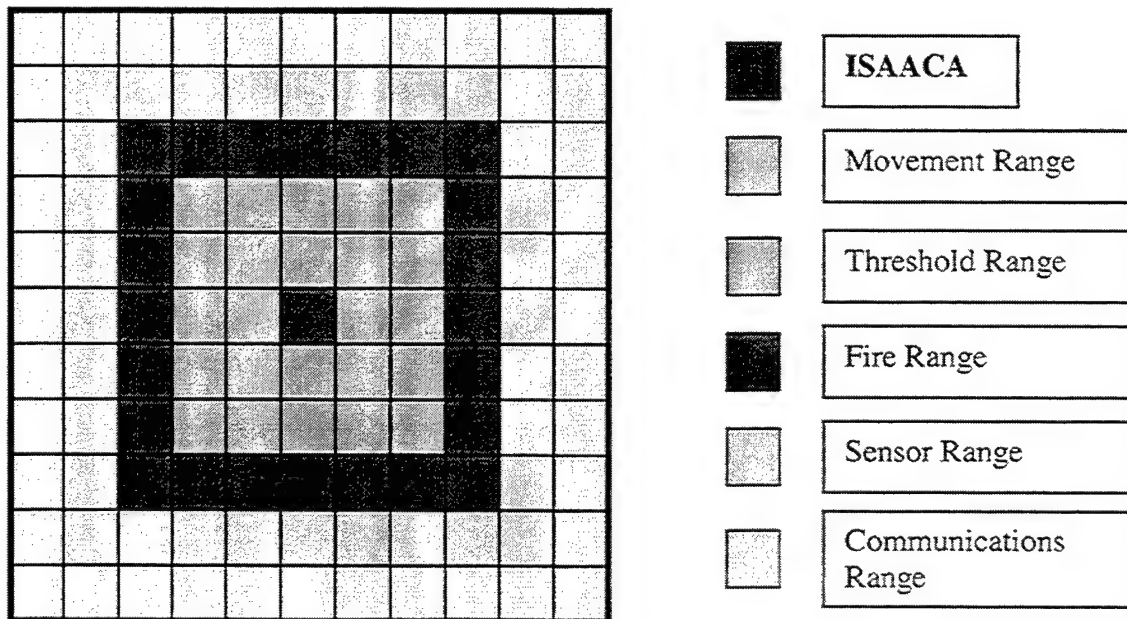
Figure 1. Various kinds of ranges that surround each ISAACA

Parameters affecting the decisions made by an ISAACA are the number of "alive" or "injured" friendly or enemy ISAACA's it can "see" and the respective distances to the ISAACA's own and the enemy flag. In addition command and control parameters may be

imposed on the ISAACAs. Each ISAACA determines where to move to based on this information. All of these parameters are rolled into one function, called the least penalty function. This function determines the ISAACA's movement.
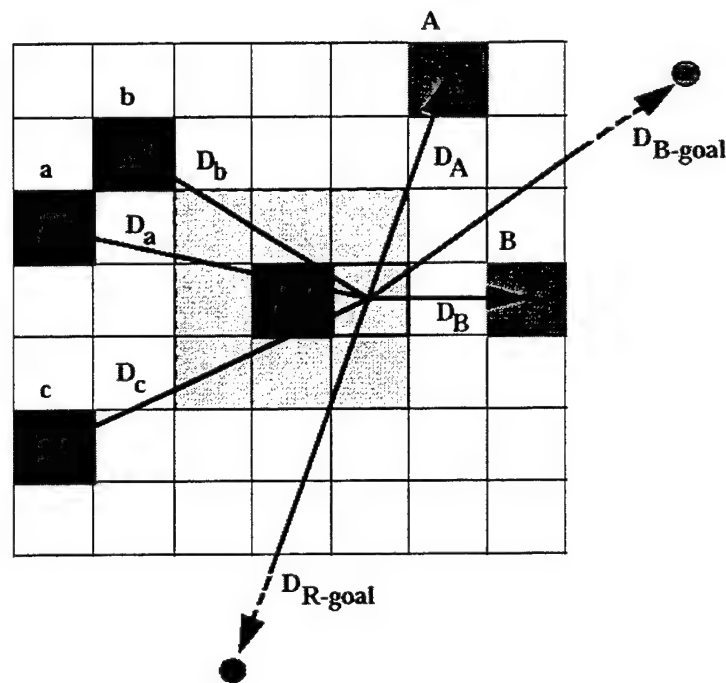
Figure 2. Sample least penalty function move calculation

In Figure 2, a sample penalty calculation is shown. The shaded area indicates the possible locations to which the agent (ISAACA) may move. The least penalty function uses the distances between the proposed location (the square from which all the arrows originate) and each agent and flag. The calculation for this square represents the value of moving from the current location to the proposed location. The agent will move to one of nine possible locations, including the current location. The location with the lowest value is the position to which the agent will move.

13

ISAAC is designed with a hierarchy of information levels. These levels roughly correspond to a chain of command, where the lowest level represents the individual soldier, tank or other single entity. Up to three levels of command (Local, Global, and Supreme) above the individual level can be instantiated in the ISAAC environment. All of these levels can issue "orders" to units below them and receive information from subordinate ISAACAs. Local commanders use the collective knowledge of subordinate ISAACAs to adjust the movement vectors of the ISAACAs. Global commanders use global knowledge to issue movement orders to local commanders. Below is a schematic with a short description of each of the levels of command in ISAAC.

Supreme Command (SC) Level
SC defines each scenario
SC defines global combat parameters
SC specifies "mission objectives"

Global Command (GC) Level
GCs define interaction among subordinate LCs
GCs use "global" (i.e. battlefield-wide) information
GCs' decisions are driven by personality-weights

Local Command (LC) Level
LC:LC interactions
LCs issue "orders" to subordinate ISAACAs
LCs use the combined "information" fields
of their subordinate ISAACAs
LCs' decisions are driven by personality-weights

Individual Combatant Level
ISAACAs respond to purely "local" information
Personality-driven ISAACA:ISAACA interactions

Figure 3. Schematic of ISAAC's hierarchy of information levels

The organization of agents in ISAAC is defined by the command structure and imposes limitations when it comes to organizing the agents in the system. The

14

organization of agents in ISAAC is fixed. That is for a given local commander, that agent can command ten other agents. So squad size cannot change. In addition once a run is set up the number of "squads" that a global commander can control is also fixed. Another way that ISAAC is organizationally limited is in that all members of a squad have the same personality or are homogenous.

Below is a screen shot of EINstien (second generation of ISAAC) running in a windows environment.
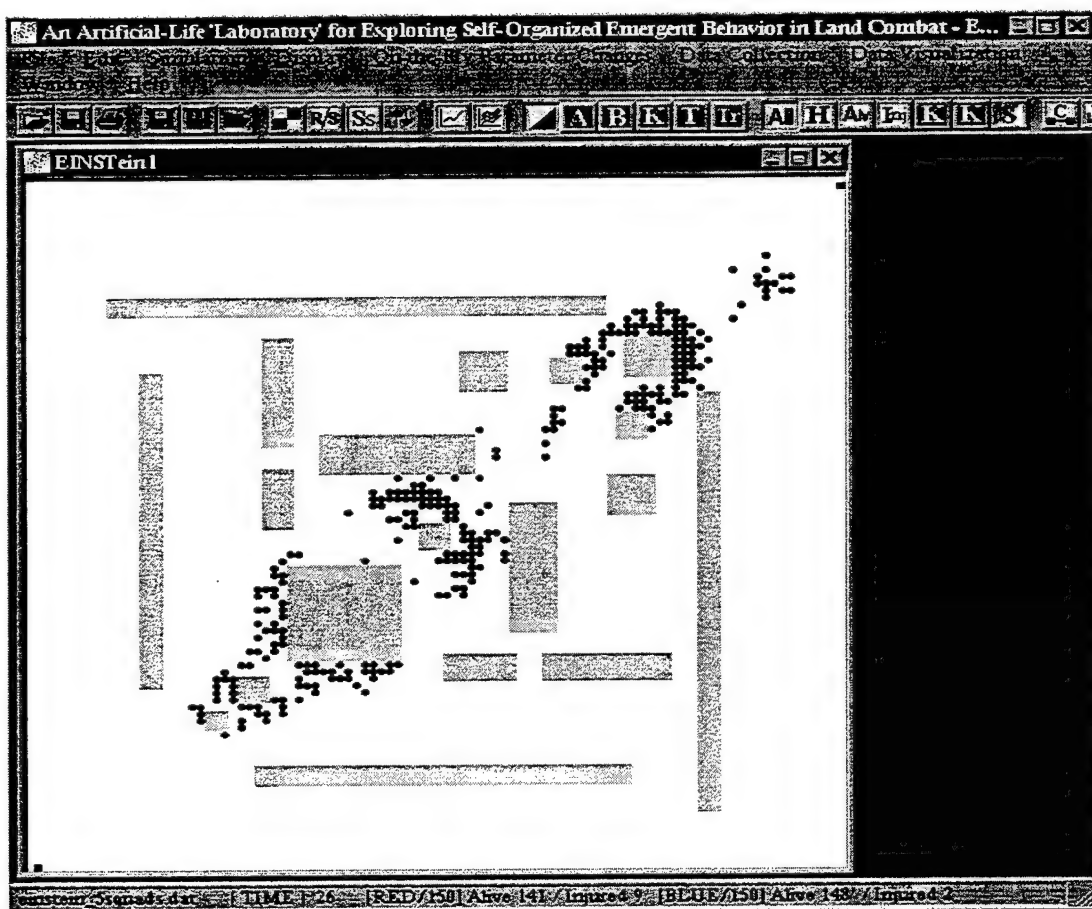
Figure 4. EINstein abstract battlefield domain

Figure 4. Above shows simulated combat between a red and a blue army each composed of five squads of ten agents each. The shaded areas represent terrain obstacles. Each army is attempting to "capture" the other's flag located in the upper right and lower

15

left corners respectively. As the agent move across the battlefield they encounter other agents. The results of these encounters are complex interactions that describe abstractly some of the fundamental concepts of land warfare. For more information on ISAAC and the follow-on project, EINSTein, see (ISAAC, 2000).

2.   Abstract Force Simulator (AFS) and Hierarchical Agent Control (HAC)

> It occurred to us that these simulators were just variations on a theme. Physical processes, military engagements, and a lot of computer games are all about agents moving and applying force to one another.   **– Mark S. Atkin**

Developed by Atkin, Westbrook and Cohen at the University of Massachusetts, the AFC and HAC are designed as a general framework for controlling agents (HAC) and a general simulator of physical processes (AFS). The two systems are designed to work in tandem as a domain general agent development toolkit.

The Abstract Force Simulator (AFS) is designed to manipulate *physical schemas* (Atkin *et al.* 1998) such as move, push, reduce, contain, block, and surround. The idea here is that moving a robot is no different than moving an army, both are instructed to move, and thus only one *move* action need be represented in the simulator. Based on this idea the AFS operates a set of objects Atkin calls "blobs". These blobs have a physical description that includes, but not limited to, mass, velocity, friction, radius, and attack strength. Each blob is capable of only two primitive actions *move* and *apply-force*. All other types of higher level actions are built from these two primitive actions.

16

Hierarchical Agent Control is a general control structure for controlling the blobs in the AFS. The physics in AFS define how a blob's actions are represented in the world; HAC defines what the blobs actions should be. HAC uses *supervenient* (Spector & Hendler 1994) architecture. This means that higher level actions provide goals and context for lower levels and lower levels return sensory information and messages to the higher levels. The *supervenient* architecture allows for the abstraction of the action process. It makes possible the building of modular, reusable actions. HAC then goes a step farther in standardizing the action-writing process into a single form.

The following diagram is an example of an HAC action hierarchy.
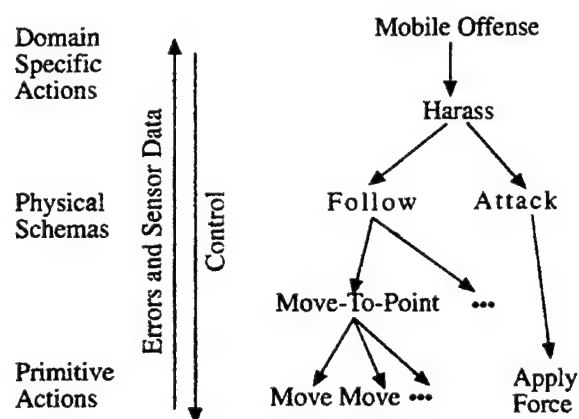
Figure 5. HAC action form a hierarchy, control information is passed down, messages and sensor integration occurs bottom-up

The test bed domain designed for the AFS and HAC toolkit was a simple capture the flag simulation. The domain was designed with terrain and two armies one red, one blue. Each army has to capture all of the other flags while simultaneously defending the flags it is responsible for from capture. The agents or blobs in each army have an action hierarchy based on the primitives *move* and *apply-force*. The action hierarchy is designed

17

to facilitate a blob moving to a location, attacking a target, defending a unit, blocking a

pass or intercepting a hostile unit. With these higher-level actions tasks for each blob are

devised such as "defend a flag" or "attack hostile unit". From these task top-level

schemas or overall goals are devised like "win-capture-the-flag".
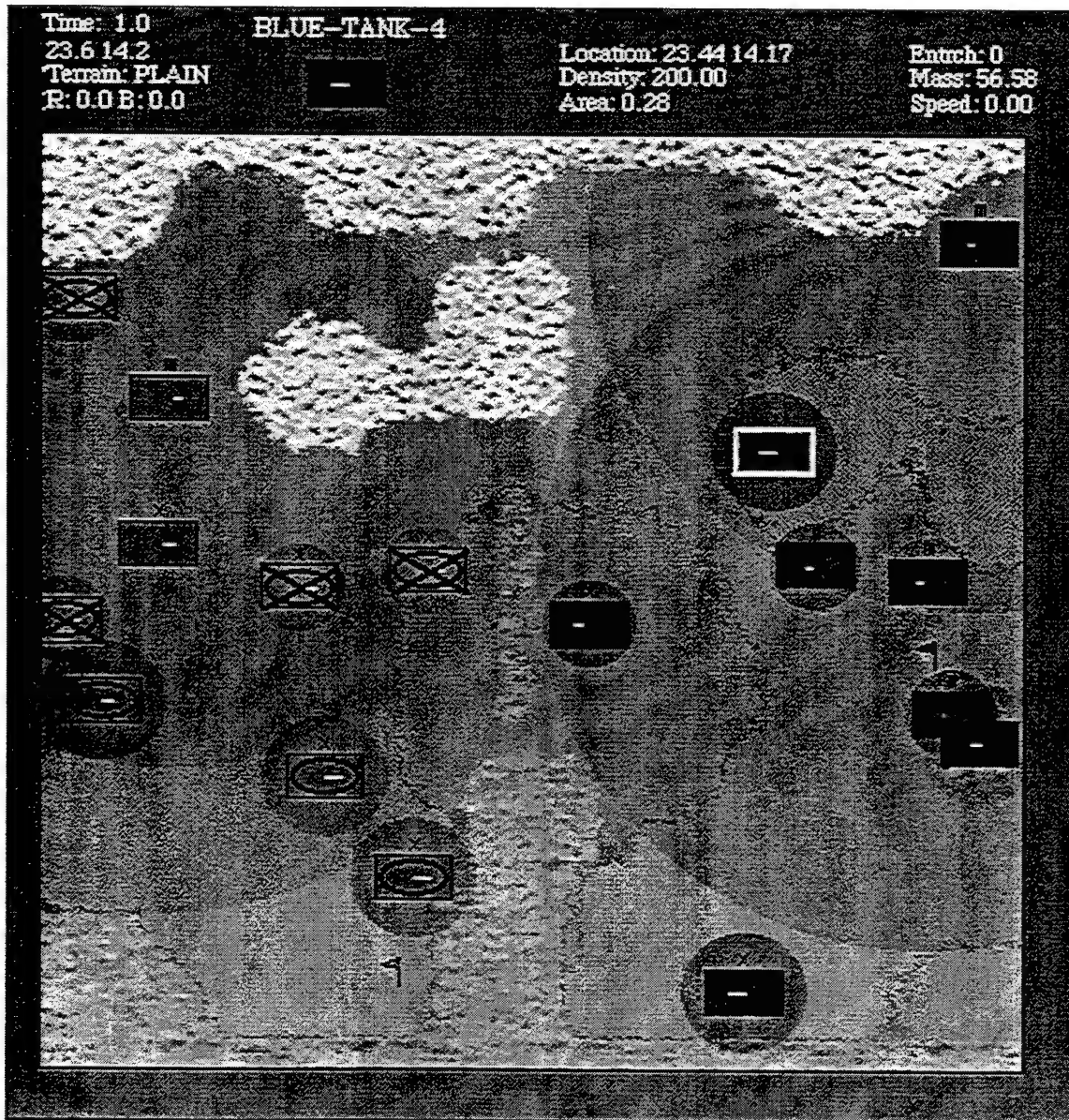
Figure 6. AFS and HAC Capture the Flag domain

D. RELATE MAS ARCHITECTURE

GI Agent is built using the RELATE design paradigm created by LCDR Kim Roddy, USN and LT Mike Dixon, USN (Roddy, Dixon, 2000).

The RELATE design paradigm is centered on using relationships between agents to define agent types or agent roles and associated goals for those agents. A relationship could be defined as soldier in an army, member of a squad or company. The analogy here is when you join the army you form a relationship with all the other members in the army. This "in-the-army" relationship has roles inherent to it. One such role would be soldier. On a battlefield a soldier has certain goals, these might be "ensure survival", "engage enemy" or "protect fellow soldier". In order to satisfy these different and possibly conflicting goals a soldier would employ a rule or one of a set of rules designed to accomplish this particular goal.

RELATE Agent Design
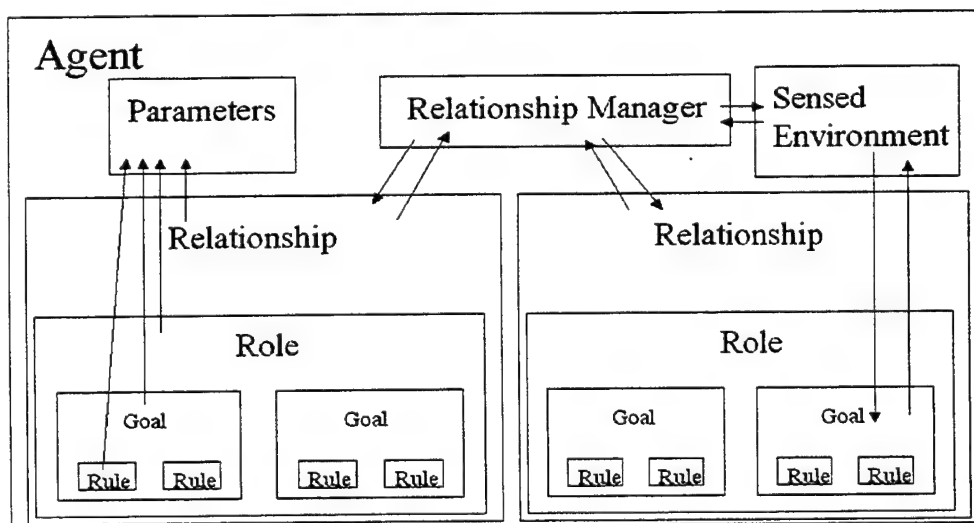
Figure 7. RELATE Agent Design Schematic

19

Figure 7. Above shows the hierarchy of the relationship, role, goal rule design structure of the RELATE agent. As shown by the diagram, an agent can have more than one relationship. Each relationship can have more than one role. Each role can have more than one goal and multiple rules per goal. After an agent establishes a relationship, and determines the role it will play in that relationship a primary goal is set. Satisfaction of this goal is determined through a feedback loop with the sensed environment, which interfaces with the MAS environment. If a particular rule is not fulfilling a goal, the feedback loop will cause the agent to choose another rule in an attempt to fulfill the goal.

See Appendix A for the design for the GI Agent Relationship/Role/Goal/Rule hierarchy.

III. GI AGENT ARCHITECTURE

A.   CHAPTER OVERVIEW

This chapter discusses the relevant primary algorithms and the design and architecture of GI Agent. After an overview of the software architecture, a detailed description of the classes and components of GI Agent follows.

The terrain dependent algorithms that are described are Line-of-Sight calculation using a ray-casting approach applied to a tile-based terrain representation and A* search to determine a path for movement, as it applies to searching the state space of tile-based terrain representation. The last algorithm to be discussed is the RELATE relationship construction as applied in GI Agent.

The GI Agent architecture model is discussed next, followed by a detailed description of the software.

B.   PRIMARY ALGORITHMS

Several algorithms from the backbone of GI Agent, these are Line-of Sight Determination, Pathfinding, and the relationship construction between the agents.

1.   Line-of Sight Determination

GI Agent uses a dynamic ray-casting algorithm to determine line-of-sight for an agent in the environment. Line-of-sight in GI Agent is calculated at each time step for each agent based on the updated sensed visual environment. This implementation is slower than reading predetermined line-of-sight for a tile from a data structure. However, generating line-of-sight data at each time step allows for dynamic, changing environment.

Although, not implemented in the version, this facilitates the GI Agents interacting and changing their environment. An example of an agent changing the terrain is the installation of obstacles, or the removal of them.

There are three pieces of data needed to calculate the line-of-sight for a given agent. First, is the sensed visual environment. The sensed visual environment is the local area of the total environment relative to the agent and within it's visual sensor range. The next piece of data required is the vertical slope from the origin (agent's location) and each of the terrain squares in the sensed visual environment. Slope is used to determine which terrain squares are considered visible to the agent at the origin. The last piece of data required is the distance from a terrain square in question to the origin.

The line-of-sight algorithm used in GI Agent is outlined below:

Line-of Sight Determination:

     Step 1. The GI Agent imports its local sensed visual environment from the environment at large.

     Step 2. Calculate the vertical slope for each terrain square in the sensed visual environment.

     Step 3. Determine the visible terrain squares in the sensed visual environment.

          a. Cast out up to 104 rays out to a range of 13 terrain squares from the origin.

          b. Along each ray, compare the vertical slope of each terrain square to the vertical slope of the terrain squares closer to the origin than the terrain square in question.

          c. If the vertical slope of a given terrain square is greater than or equal to the vertical slopes of all the terrain squares between it and the origin then the terrain square in question is visible from the origin.

Below is a diagram of an agent and a sample of the line-of-sight rays with corresponding terrain squares.
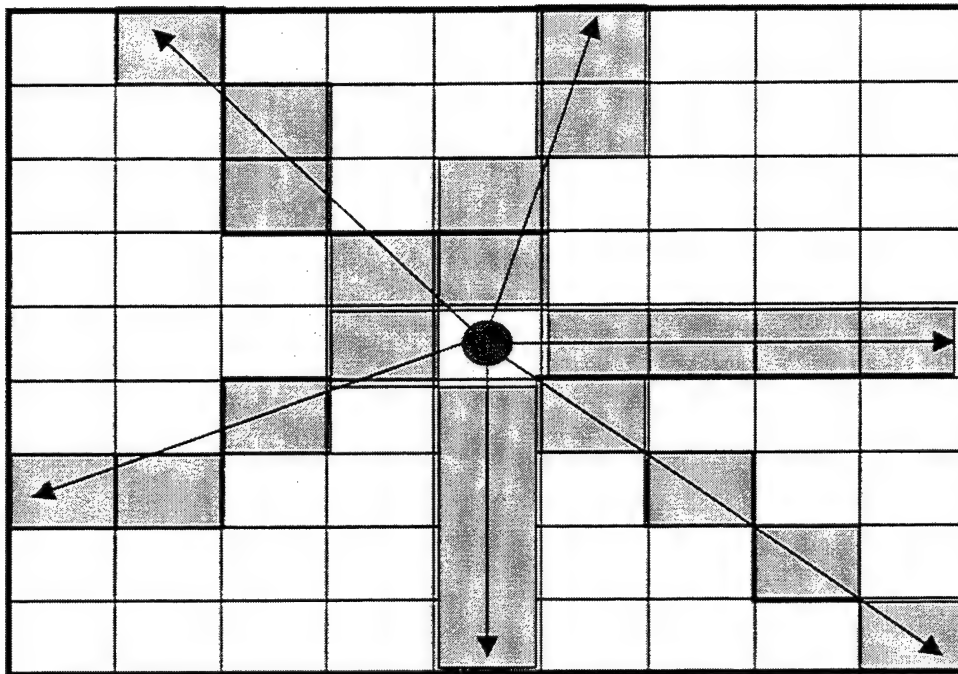
Figure 8. Line-of-Sight Ray Casting

The above figure shows the individual rays cast out from the agent and the specific terrain squares affected by each ray. This diagram is only a sample of the 104 rays cast to determine line-of-sight for each agent. Terrain squares that lie along a particular ray and are not considered by that ray are covered by adjacent rays. This is done to keep the aliasing effects ray casting on the discrete squares to a minimum.

Below is an example of line of sight showing visible agents and non-visible agents. The blue GI Agent in the center is the agent who's line-of-sight is being represented. GI Agents with the large X over them are not visible to the GI Agent in the center.

The white area is level open ground, the green area is wooded and a GI Agent can only see one square in to the wooded area. The tan area represents an increase in

elevation but still open ground. The brown area is a second increase in elevation also still open ground.
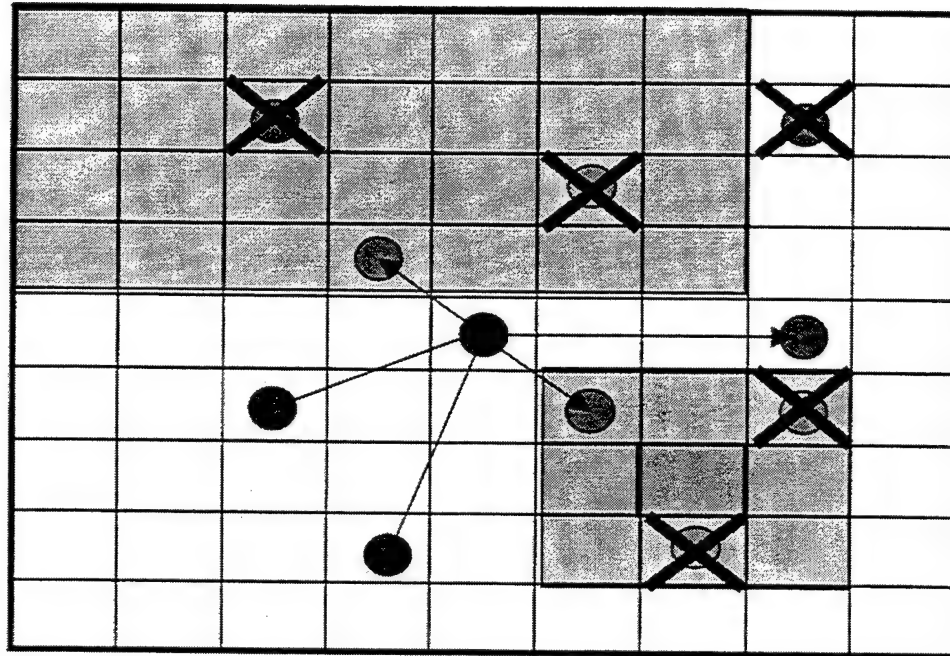
Figure 9. Line-of-Sight Example

The blue agent in the center of the figure can "see" the other agents that have arrows pointing to them, but not the agents that have large a X over them. Although the blue agent in the center and the agents in the green wooded area are on the same level, the line-of-sight algorithm takes into account the vegetation and only allows the agent to see one square into the wooded area. The blue agent in the center can only see one red agent on the hill because the LOS algorithm only allows an agent to see the squares on the edge of the hill facing the agent in question

The next figure below shows a blue force GI Agent (represented by the yellow agent) line-of-sight calculation; the area in gray is the non-visible portion of the agent's sensed visual environment.
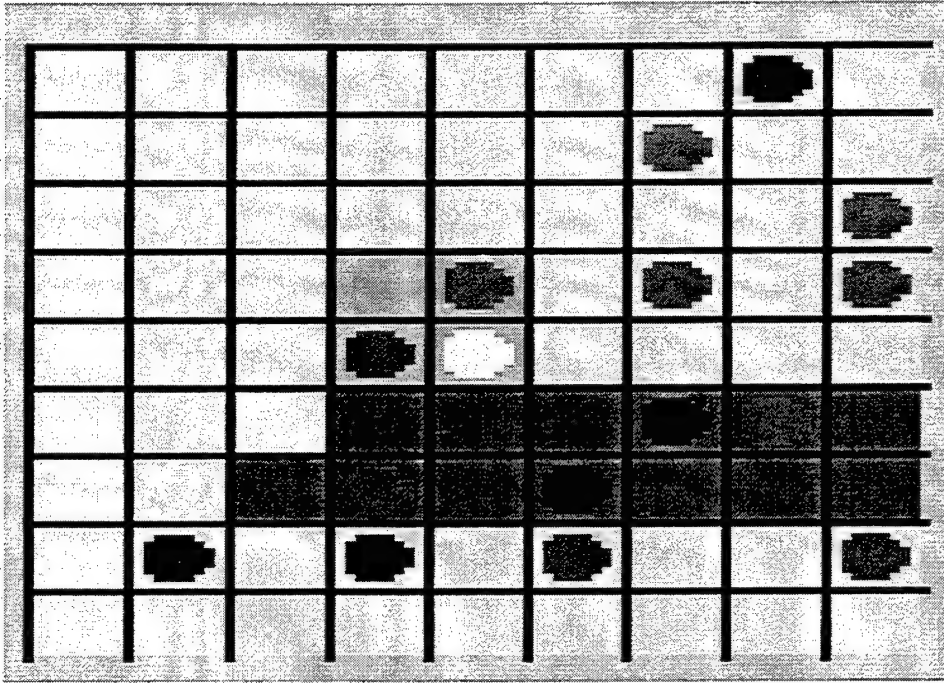
Figure 10. Sensed Visual Environment

2. A* Search for Pathfinding

For terrain, navigation or path finding on a terrain model based on squares A* search is a natural fit for finding the best path using a given set of movement criteria.

A* search is superior to depth-first search, breath-first search, and best first search. A* search versus depth-first and breath-first search, A* will produce a better path in significantly less time. Both depth-first and breath-first will produce a path that get to the goal, but neither algorithm has the ability to evaluate the generated path for fitness in regards to a heuristic evaluation function. Best-first search is capable of discovering the optimal path. However, best-first cannot backtrack, these results create longer search times and sometimes non-optimal path generation.

The basic capabilities of the A* search are directional search, backtracking, and path variation based on a heuristic function. The result is fast search times, basically O

25

(N). Using the right data structure A* search times can be as low as O (1). The implementation of the algorithm in GI Agent, search times are close to O (N) based on distance from goal.

Directional search in path finding may seam obvious but when the algorithm is used for other applications, it may not be obvious how to determine direction. This is not a problem in path finding, determination of a direction is simple coordinate comparison. The result of the coordinate comparison is to reduce the number of nodes searched at each level by fifty to sixty three percent depending on how the terrain is implemented. Directional search of a path four terrain squares long results in reducing the number of possible nodes that are to be searched from a total of 4680 to a pool of 120. From the pool of 120 only 12 are actually searched, a reduction of 4668 nodes that need to be searched.

Backtracking is the ability of an algorithm to remember where it has been and be able to return there, and then take a new course from that point. This is similar to taking a road to a destination. Arriving at a fork in the road. One direction is tried when that starts to go in the wrong direction or dead-ends, back tracking to the fork and taking the other direction to the destination. In a dynamic environment, the ability to adjust a path in this manner is crucial as obstacles may appear along a path after the entity has computed the path. Back tracking allows the entity to find a new route to its destination.

A heuristic function is used to evaluate the fitness of the path. This function is designed to give the entity the ability to choose the best path for a given environment. The parameters of this function are manipulated by the entity, the environment or by other entities. Modification of the heuristic function results in different paths to the same

26

goal for a given environment. Heuristic function parameters that could be used are terrain

elevation changes, terrain cover or concealment, enemy or friendly forces; movement

cost of the terrain, or internal factors in the entity. This is only a sample other factors are

possible.

This makes A* pathfinding a natural fit for rule based movement, as each rule

could simply modify the inputs to the A* heuristic function allowing for a great variety of

movement styles.

The major steps to the A* path finding algorithm are:

A* Path Finding:

    Step 1. Determine direction of travel to goal.

    Step 2. Evaluate nodes that correspond to direction of travel.

        a. Check if current node is blocked.

        b. If blocked node, back up one step.

        c. Evaluate each possible node.

        d. Add best node to the path.

    Step 3. Determine new direction of travel.

The figure below shows the best-case scenario for path finding using the A*

algorithm with directional search. From the staring node the only nodes that are searched

are the nodes that can lead to the goal. If the path does not encounter any obstacles the

path plotted by the algorithm will be the shortest distance path to the goal.
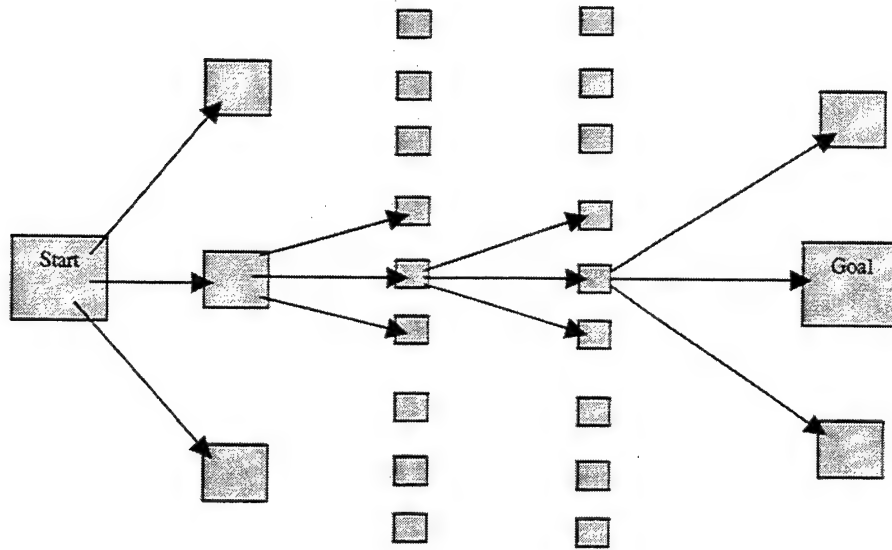
Figure 11. A* Search Minimum Distance Path
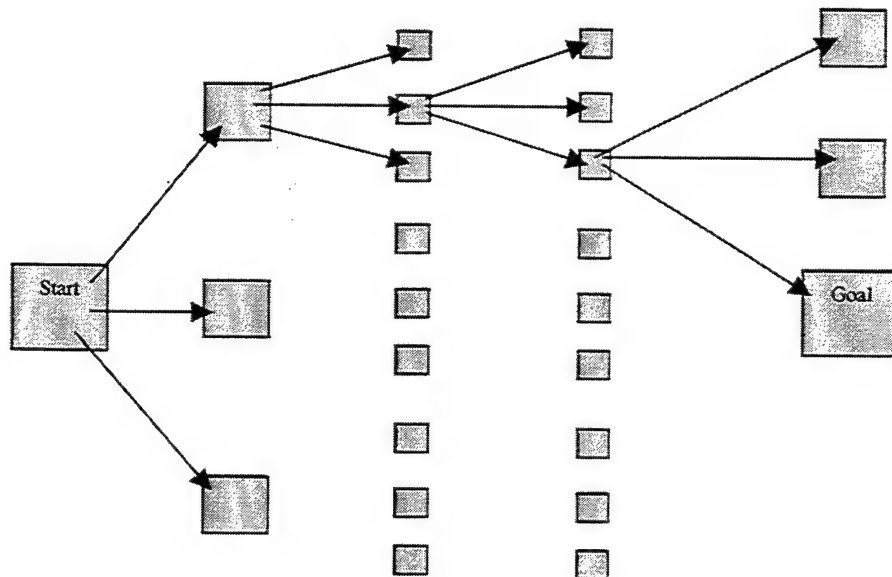
Figure 12. A* Search Alternate Path

The above figure represents an alternate route using the A* path finding

algorithm. From the start node the path search encounters an obstacle in the direct path to

the goal. The A* search then chooses the shortest route around the obstacle that goes to the goal

3. RELATE Relationship Construction

RELATE has a built in algorithm for construction of relationships between agents. This algorithm has several shortcomings when applied to the formation of relationships in GI Agent.

The organizational structure of agents in GI Agent is designed to replicate a dismounted infantry company. This provides a clearly defined set of relationships, with clearly defined roles. RELATE however, does not take into account a military chain of command. In RELATE a squad relationship is the same as any other squad relationship. The RELATE algorithm alone produces squads of varying size and in different positions. This produces a company organized in a haphazard and random fashion. A real infantry company each squad in each platoon is a separate and different squad. The squads and platoons generally are the same size, with the same organizational makeup. In order to achieve a consistent company organization a new paradigm for organizing the company was required.

The solution is to assemble each platoon in an assembly area, with squads forming in sub-assembly areas. This way only the agents that should be in that particular platoon and squad join that platoon and squad. The assembly area paradigm can be easily modified to fit the structure of almost any organization, given enough space. The figure below shows the layout of a standard company with three platoons of three squads each. The company is laid out in a fixed formation with the platoon ordered from north to

29

south. The squads are integrated in the larger platoon assembly area, in a triangular formation. The platoon assembly areas in GI Agent will hold up to four squads. The squad assembly areas are fixed in size and can hold up to 15 soldiers. A drawback of this arrangement is that it limits the variety of organizational structures that can be modeled in GI Agent.
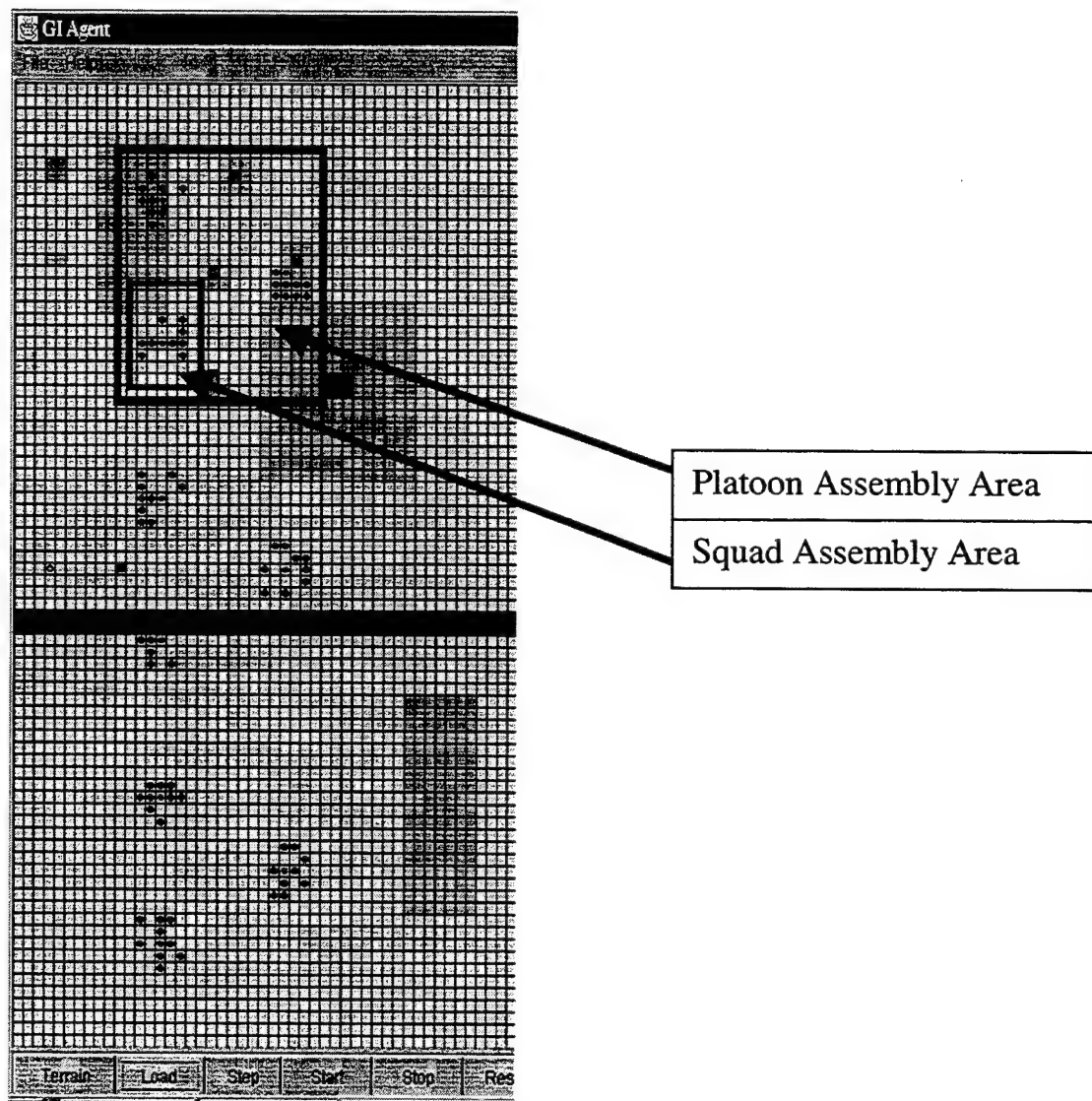
Figure 13. Platoon and Squad Assembly Areas

The RELATE paradigm for creating a relationship between two agents is for the agent to be able to sense each other. The original RELATE agents are restricted to one sensed environment. Hence, there is only one way for an agent to communicate to another agent that it wants to form a relationship with. This defines the second problem, how to form a relationship between two agents that cannot sense each other.

The solution was to modify RELATE to allow an agent to have any number of sensed environments. In GI Agent, each agent has two sensed environments, a visual and communications sensed environment. The visual sensed environment represents the local area that the agent can "see". The communications sensed environment represents the mental environment that every soldier has when communicating with another soldier over a radio. The two soldiers are out of sight of each other but obviously aware of each other. The radio communication between a company commander and his platoon leaders separated by terrain is an example of this type of sensed environment.

C.   GI AGENT DESIGN

1.   Software Architecture

The overall design structure of GI Agent uses a java-based class GIAgentSim to house the environment class, GIAgentSimEnv. The primary functions of the GIAgentSim class are to start the simulation and interface between the SimEditor and the environment.

The primary functions of the GIAgentSimEnv class are to contain the terrain, agents and classes required to interface between them. Below is a diagram of the basic structure of GI Agent.
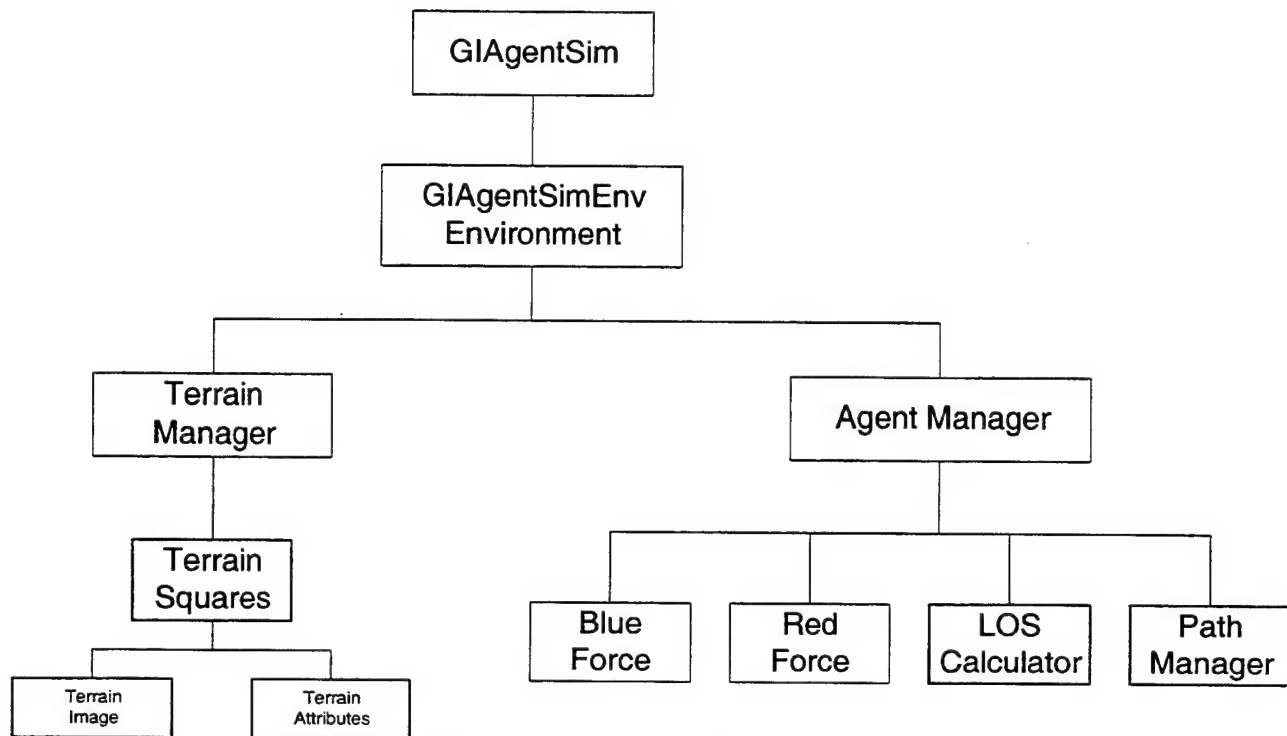
Figure 14. GI Agent Design

The above figure shows the relationship between the major classes of GI Agent. The GIAgentSim class contains two major classes the SimEditor Graphical User Interface (GUI) and the environment, GIAgentSimEnv class. The SimEditor is the user interface provided to allow the user to set the parameters of a simulation run. The GIAgentSim class passes the simulation parameters into the environment. The Terrain Manager and the Agent Manager interface with each other through the GIAgentSimEnv class.

a. Terrain

The Terrain Manager is primarily responsible for the management of the individual terrain squares and reading out terrain data to the GIAgentSimEnv class. A breakdown of the Terrain related classes are show below.
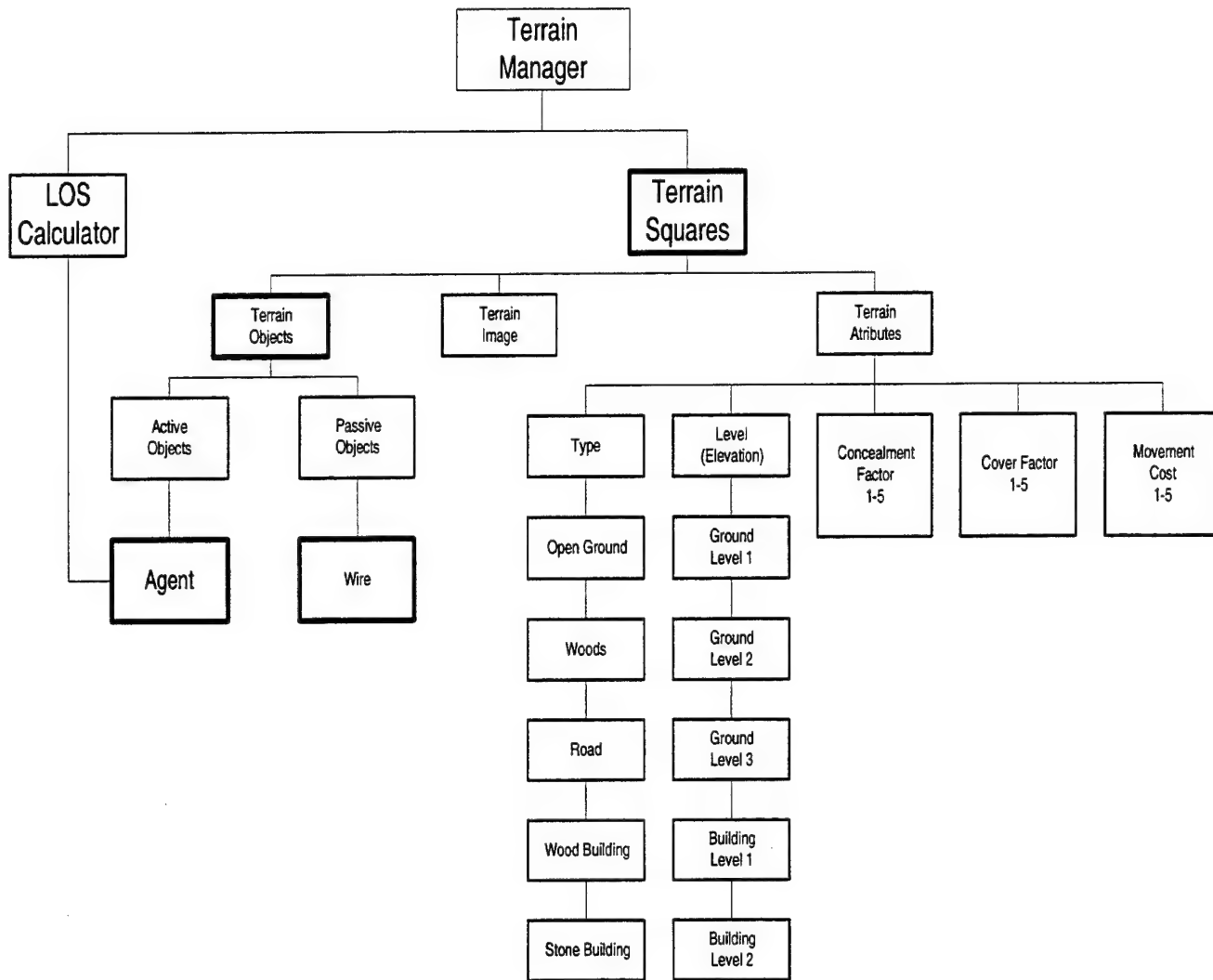
Figure 15. Terrain Manager and related classes

The above diagram describes the connections between the various classes that work with the terrain model in GI Agent. The Terrain Manager contains a two dimensional array consisting of Terrain Squares. The Terrain Squares contain the listed attributes, an image if available and if present an agent. The agents do not directly interface with the terrain squares but receive terrain data through the agent manager by means of the LOS (Line-of-Sight) Calculator. The LOS Calculator interprets the terrain

33

data by means of the algorithm explained earlier in this chapter and presents the data to the agent.

The Terrain Square is the basis for the terrain model in GI Agent. Five types of terrain and three levels of elevation are modeled. The terrain was kept simple, as the scale of the model is five meters per square, roughly the minimum distance between two infantrymen in modern combat.

Water

Road

Open Ground Elev 1

Open Ground Elev 2

Open Ground Elev 3

Woods Elev 1

Woods Elev 2

Woods Elev 3

Wood Building

Stone Building
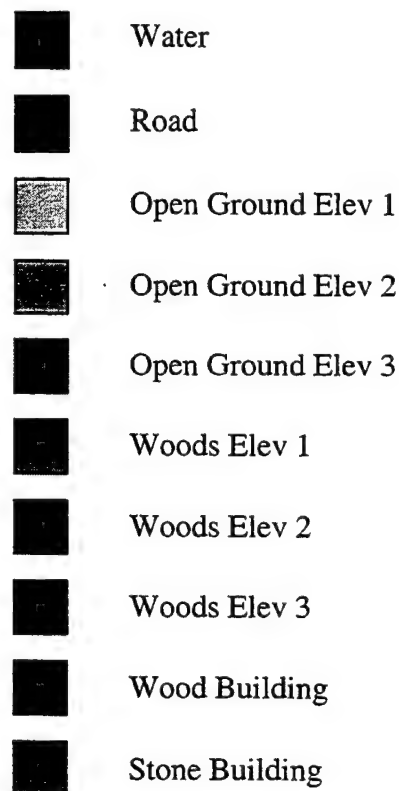
Figure 16. Terrain Square Key

There are two dialog boxes intended to interact with the terrain in GI Agent. The first is the Square Dialog Box. The dialog box gives parameter data on a terrain square and allows for the modification of an individual terrain square. The second dialog box is the Terrain Block Dialog Box. This dialog box enables the user to select a

"block" of terrain and modify it as desired. To activate the dialog box left click at a start point and drag to the opposite corner of the selected terrain block. Release the mouse button and the dialog box will appear.

Figure 17. Terrain Square and Terrain Block dialog boxes

b. *Agent Manager and agents*

The soldier is the primary and most powerful mechanism of war.
Jose Vilabla, Spanish General and Historian

The Agent manger is housed in the GIAgentSimEnv class and is the conduit of information from the terrain to the agents and run cycle manager. The class sets up the initial formation of the agents and passes in parameters from the user interface to the agents. Then agent manager class starts and maintains the run cycle of the agents. In addition to the agents themselves, Line-of-Sight and path finding classes are contained in the agent manager.

GIJoelAgent is the class that forms the basic structure of an agent in GI Agent. GIJoelAgent extends the relate Agent and adds in numerous parameters for sensing, moving, shooting, communication and identification. The default capabilities of a GIJoelAgent are a sensing range of six squares, a shooting range of four squares and a movement range of one square per time step. In addition, each GIJoelAgent can fulfill several roles depending on the relationships it forms. These are rifle soldier, sniper, squad leader, platoon leader and company commander.

The figure below shows the "under the hood" parameters of a GIJoelAgent. The interface called the Brain Lid allows a user to examine the parameters and decision-making processes of an agent. The brain lid details the agent's personality and formed relationships in the upper left quadrant of the dialog box. In the upper right quadrant is the agent's sensed visual environment, with the agent in question colored yellow. The lower half of the dialog box shows the movement goals with current rule and the combat goals with the current rule.

Figure 18. GI Agent Brain Lid

The life cycle of an agent consists of two major parts, the initial creation and the run loop. The creation of an agent sets the initial agent parameters, assigns the agent to a unit and establishes the initial relationship set for the agent. The run loop is the time-step-by-time-step management of an agents actions and decisions.

The run loop consists of six major events in a time-step for an agent. The first event is the agent had the current sensed environments loaded into its sensed environment array. Next, based on the new information in the sensed environments the agent checks if it can form any new relationships. Once any new relationships are formed the agent then polls all its goals to assign a current goal. Credit is assigned to all the goals

37

and associated rules based on the current sensed environments and input from the agent's chain of command. The agent now has a goal to try to achieve and rules or set of rules that will enable achievement of that goal.

The last three steps in the run cycle are communicate, shoot, and move in that order. The communicate step has the agent report to its direct superior certain information about itself and it's environment. For a squad leader it's the average health of the members of its squad and the current total of enemy forces in contact with the squad. During the shoot step the agent uses the current engagement goal and rule set to select a target agent and shoot at it. The final event in the run cycle is the move event. The agent uses the current movement goal and rule set to determine the movement path for the agent. The agent then moves one square along that path.

c. *Path Manager*

Path Manager class calculates the desired movement path for an agent based on the sensed visual environment, overall movement goal and the current movement rule. Determination of a path segment is limited to the area of the sensed visual environment of an agent. An agent may have 100 squares to travel to get to it's movement goal and a visual sensor range of six, the result will be that the path manager will only calculate the next six steps towards the goal. An agent knows that its goal is in a direction but can only plan the path to get there for as far as it can "see".

Agent paths are determined using an A* search algorithm described earlier in this chapter. Path Manager uses a weighted value heuristic function and a current movement goal to determine a specific path. The current movement rule passes in the

38

values to be used in the heuristic function and possibly a local movement goal. Through

the manipulation of these parameters that different movement behaviors are achieved.
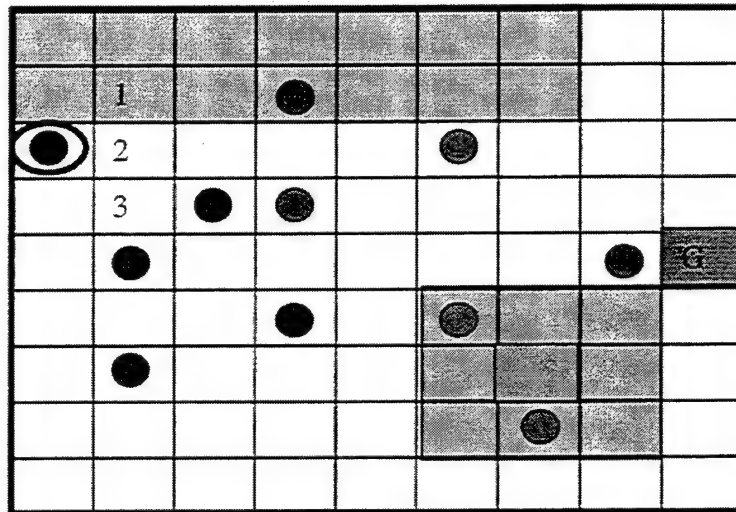
Figure 19. Path Calculation

The diagram above shows an example calculation of a path segment. The

circled Blue agent on the left side of figure 9 is moving toward its goal on the right side

of the figure. The agents movement direction is east and as such it will chose its next

move from one of the three labeled squares to the east of it. The square with the lowest

heuristic value will be the square chosen by Path Manager for he agent to move into next.

The sample calculation is based on a movement rule that places a priority on maintaining

a covered and concealed route as much as possible, while ignoring movement cost. This

is only one of many possible movement rules. As show by the calculations in the figure

below the agent may not take the most direct route to an objective. In this case the agent

selects the wooded terrain square to move into next actually moving away form the

agents stated movement goal.

The calculation and results for each square is detailed below.

Square 1 Calculation Parameter	Parameter Value	Weight Factor	Adjusted Value
Goal Distance	9.48	0.8	7.589
Way Point Distance	0	0.4	0
Elevation Delta	0	0.6	0
Agent Count	0	0.2	0
Cover	1	1	1
Concealment	1	1	1
Movement Cost	1	0	0
		Square Value	9.589

Square 2 Calculation Parameter	Parameter Value	Weight Factor	Adjusted Value
Goal Distance	9.21	0.8	7.375
Way Point Distance	0	0.4	0
Elevation Delta	0	0.6	0
Agent Count	1	0.2	0.2
Cover	3	1	3
Concealment	3	1	3
Movement Cost	1	0	0
		Square Value	13.575

Square 3 Calculation Parameter	Parameter Value	Weight Factor	Adjusted Value
Goal Distance	9.055	0.8	7.22
Way Point Distance	0	0.4	0
Elevation Delta	0	0.6	0
Agent Count	2	0.2	0.4
Cover	3	1	3
Concealment	3	1	3
Movement Cost	1	0	0
		Square Value	13.644

As the calculation shows the agent's next move is into square one. The ability to easily manipulate a route in this fashion allowed for numerous different movement rules to be created and used.

There are seven parameters in the heuristic function used to select the next movement square. Goal distance is the distance from the agent to its overall movement

goal. Waypoint distance is the distance from the agent to its intermediate movement goal, if it has one. Elevation delta is the difference between the elevation at the agent's location and the elevation at the proposed movement location. Agent count is the number of friendly agents that are within one square of the proposed movement location. Cover, concealment and movement cost are terrain attributes of the proposed movement square.

d.   *GI Agent MAS Simulation Editor*

The Simulation Editor dialog box is the interface for the user to set the parameters for a simulation run. There are three main components of a simulation run, the mission and organization of the blue and red forces, the attributes of the individual agents, and the length of the run.

Mission and organization of the forces is set using the slider bars and lists in the Organization block of the editor dialog box. Function of each interface tool is listed below:

Squad Elements Slider Bar: set the number of rifle soldiers in a squad from 3 to 15. Maximum squad size possible to include snipers is 15.

Platoon Elements Slider Bar: set the number of rifle squads in all platoons from 2 to 4.

Company Elements Slider Bar: set the number of platoons in the company from 2 to 4.

Number of Snipers Slider Bar: set the number of snipers to place at a level of an organization. i.e. selecting one on the number of snipers slider bar and squad on the sniper level list, results in one sniper being added to each squad for a total of nine snipers in a default company of three platoons with three squads each.

Sniper Level List: set the level of organization at which snipers are to be integrated into the unit.

Mission Selection List: Chose one of available missions for the company.

Mission Descriptions:

Attack mission –Force will attempt to secure opposing force colored square on opposite side of terrain board.

Defend mission- Force will attempt to defend the like colored square on same side of terrain board.

Recon mission- Force will move across the terrain board until enemy contact is made, then the force will attempt to destroy the enemy force.

Length of Simulation Run Slider Bar: set the number of times to run the simulation, in increments of ten.

Agent abilities are the combat parameters for the soldiers in GI Agent.

Sense Range Slider Bar: set the radius of the sensed visual environment for the rifle soldier agents, sniper sensed visual range is double the range of the rifle soldiers. Default range is six, twelve for snipers.

Weapons Range Slider Bar: set the range of agents weapon, sniper's range is double that of the rifle soldier weapons. Default weapons range is four squares for rifle soldiers, sniper default is eight squares.

Probability of Hit Slider Bar: set probability a given shot by an agent will hit another agent, without modification. Default probability of hit is 0.5 for rifle soldier and 0.9 for snipers.

Durability Slider Bar: set the number of times an agent can get "shot" before it dies.

Update Force Button: Based on the settings of the organizational slider bars click the update button to calculate the number of agents in a given force. Both forces can be calculated simultaneously.

Start Simulation Button: click to set the chosen parameters and start the simulation.
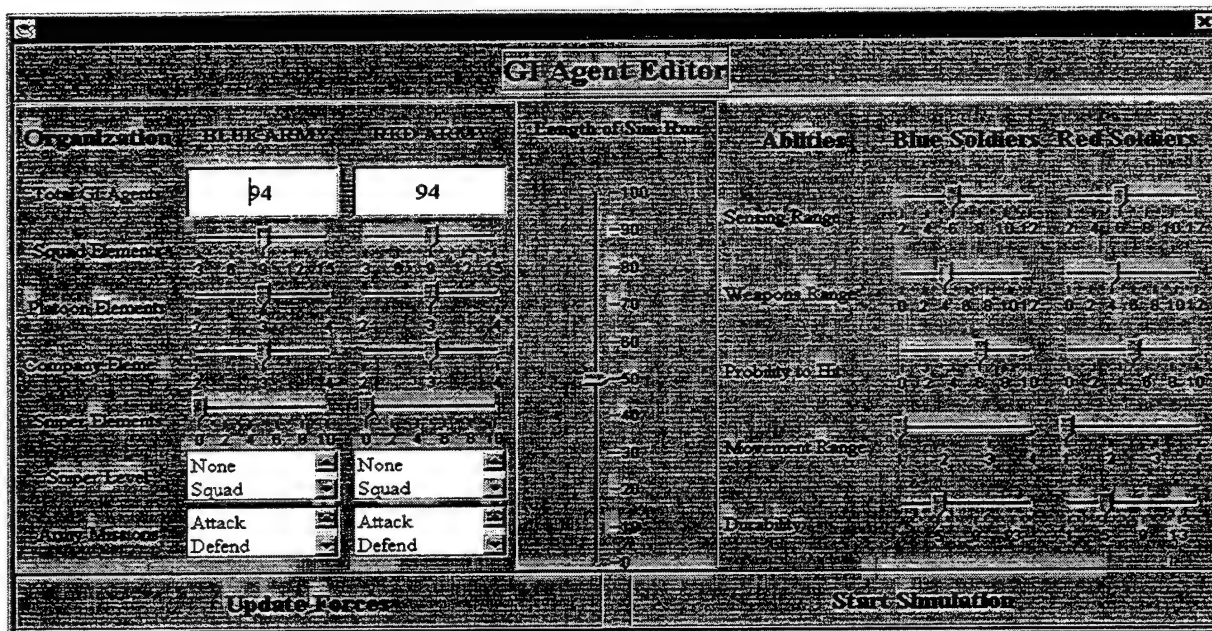
Figure 20 GI Agent Simulation Editor Interface

THIS PAGE INTENTIONALLY LEFT BLANK

IV. GI AGENT ORGANIZATIONAL EXPERIMENT

A.   CHAPTER OVERVIEW

This chapter discusses the design of the organizational experiment and results of the experiment. The organizational experiment is designed to provide a foundation of relevancy for the multi-agent system GI Agent. The experiment is not meant to prove the value of one organizational design over another, but to provide insight into and understanding of the complex environment of light infantry combat. It is not possible in the scope of this thesis to investigate all the possible combinations of unit organizations, agent capabilities and terrain models, as the combination of variables is immeasurable.

B.   GI AGENT ORGANIZATIONAL EXPERIMENT DESIGN

1.   Purpose

The purpose of this experiment is to discover any possible differences in unit effectiveness as a result of changing the organizational structure of a unit. The type of unit involved in this experiment is a dismounted light infantry company augmented with snipers.

The justification for analyzing the data produced by GI Agent is quantifying any insights gleaned from the MAS simulation. In other words, the data produced actually means something and is not just a collection of random numbers created for my own entertainment. The overall effectiveness of the infantry company organization is evaluated not the individual parts. The data produced by GI Agent is analyzed by comparing the averages of several different measures of effectiveness. The primary

45

measure of effectiveness (MOE) used was the average Killed in Action (KIA) of the company organization in question.

2. Method

This section describes the area of exploration, agent profiles, terrain model and unit organizational structures.

a. Area of Exploration

The experiment conducted consists of three unit mission scenarios. These are Attack, Defend, and Movement to Contact. In each of the three scenarios three different infantry company organizational structures are measured against an equal sized infantry company without sniper augmentation. The same terrain model is used in all three scenarios. Agent's capabilities and personalities are identical among all agents of both forces, with the exception of the Sniper agents, which have approximately twice the combat power of the riflemen agents.

b. Unit Organizational Structures

The three different treatments applied to the organizational structures of the units are snipers attached at the squad, platoon and company level. In the squad level treatment one sniper is attached to each squad in the company and under the control of the respective squad leader. In the platoon treatment, a three sniper section is attached to each platoon and under the control of the platoon leader. In the company treatment, all nine snipers form their own squad and operate under the control of the company commander.

Diagrams of each of the three infantry company organizational structures are shown below. Sniper organizational positions are highlighted in bold outline.
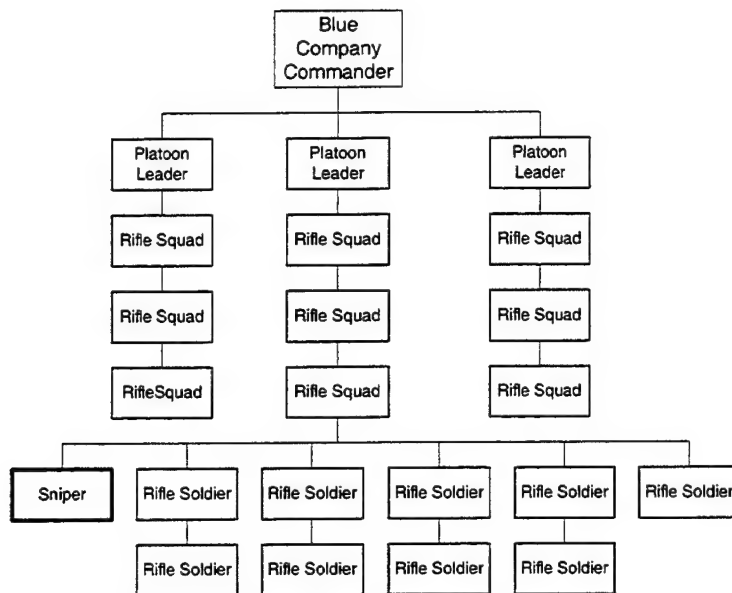
Figure 21. Company Structure with Snipers Attached to Squads
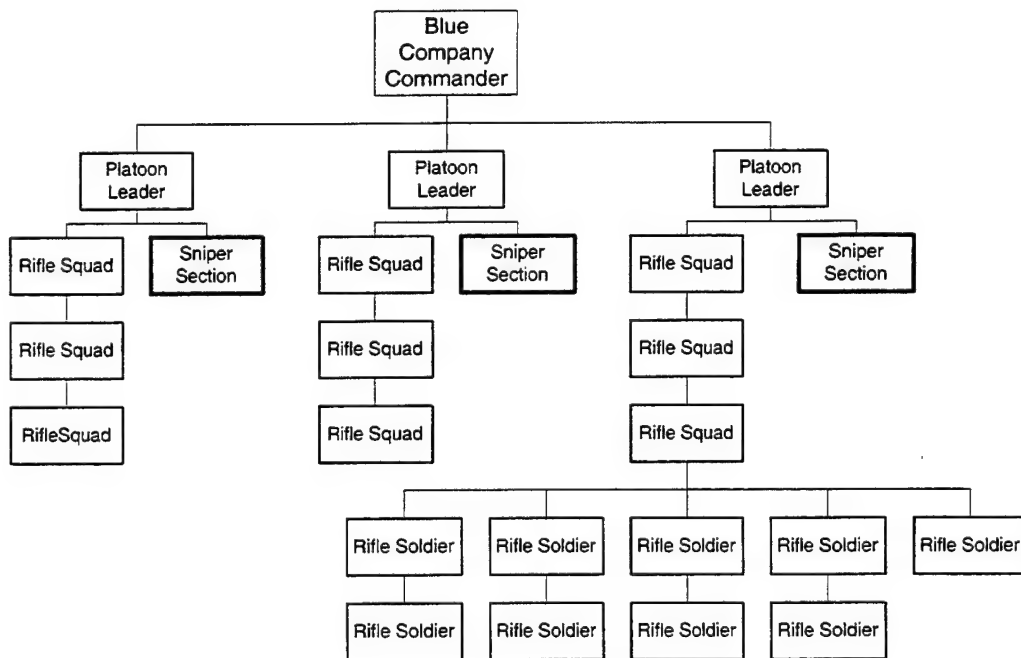
Figure 22. Company Structure with Sniper Sections attached to Platoons

47
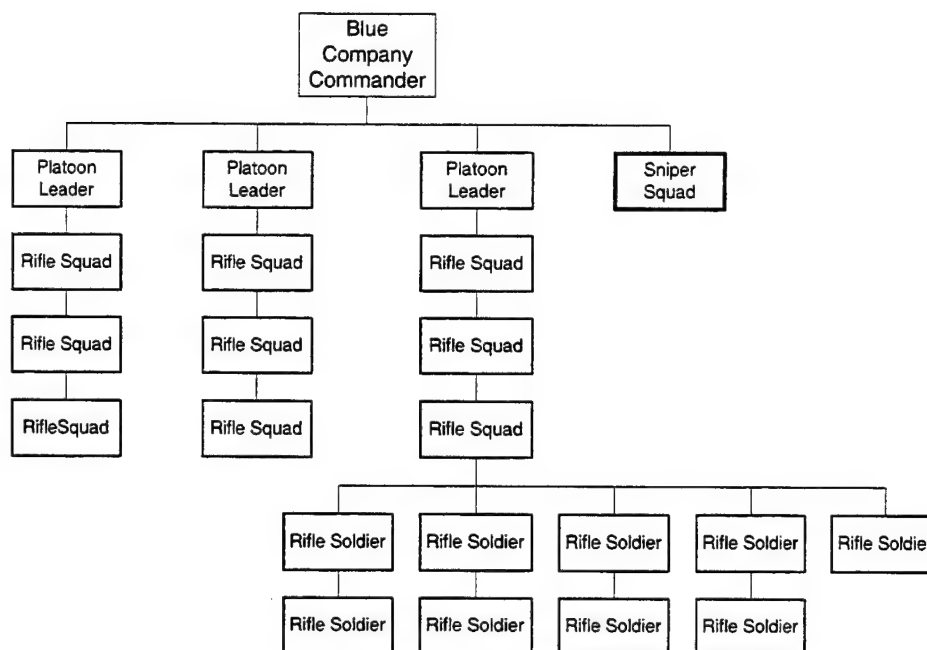
Figure 23. Company Structure with Sniper Squad attached to Company

The base treatment infantry company consists of three platoons of three squads with ten rifle soldiers in each squad for a total of ninety-four soldiers. Nine snipers are added to this base company for a total of one hundred three soldiers in a treatment company.

The opposing force infantry company is nearly identical to the infantry companies described above. The opposing force infantry company consists of three platoons of three squads of eleven rifle soldiers each. This results in a total of one hundred three soldiers in the company including the company commander and platoon leaders. An additional rifle soldier is added to all the squads in the opposing force company to provide the company with an equal number of soldiers.

c. *Terrain Model*

A single terrain model was used for all three experiments, although different portions of the terrain model were used in each experiment. The basic design of the terrain was to allow the agents to interact with the terrain but not to allow the terrain to overly influence the results of the experiment. The purpose being to study the affects of changes in organizational structure on unit effectiveness and not the influence of terrain. With this in mind the terrain was designed with large open areas interspersed with wooded ground, wooded hills, open ground hills and a few buildings. One small water feature was also added.

The terrain model is shown below.
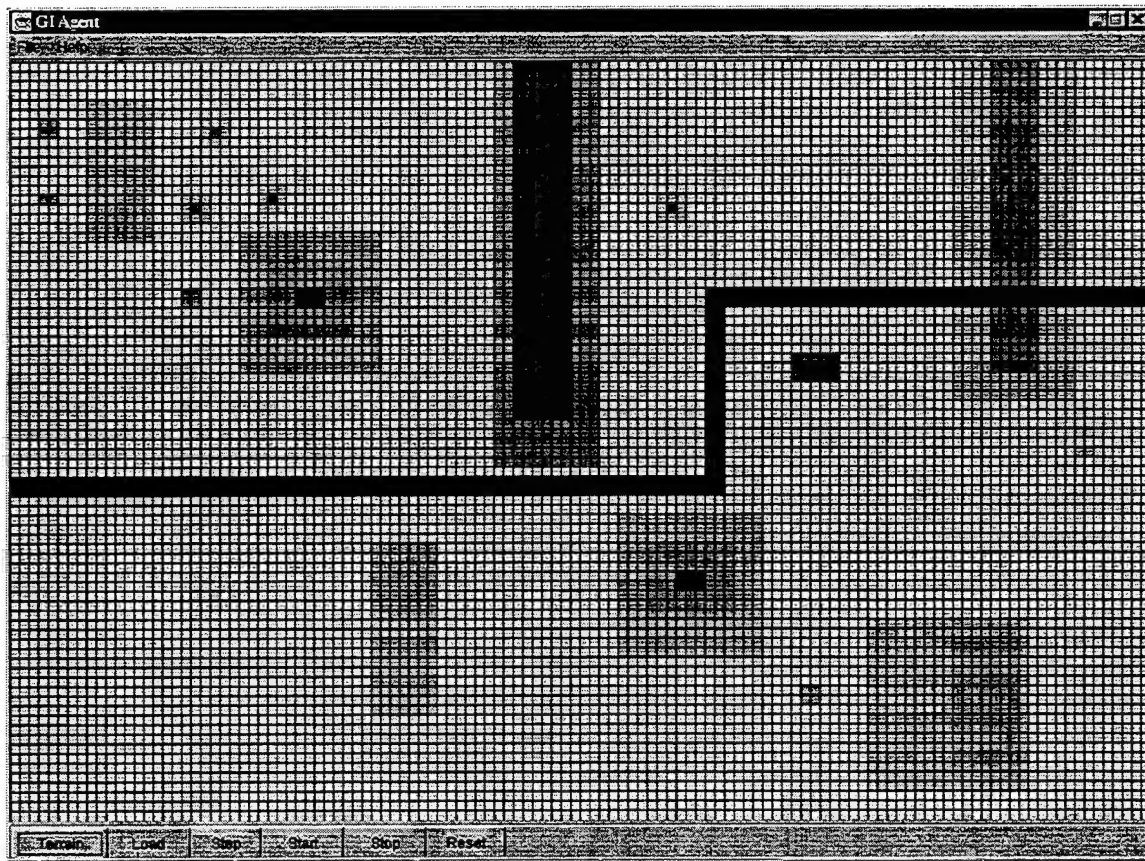
Figure 24. Terrain Model

49

d.   *Agent Profiles*

Each agent has a list of personality traits and ability parameters. Personality traits are uniform for all agents and detailed in the table below. This makes the threshold for various agent actions equal among all agents. The reason for uniform personality traits is to remove the affect of various agent personalities on unit integrity. Personality traits modify agent goal selection. For example an agent with a high self-preservation is more likely to select its "ensure survival" goal with less input from the environment.

GI Agent Personality	
Traits	**Factor**
Independence	0.25
Aggressiveness	0.8
Self Preservation	0.2
Loyalty	0.2
Obedience	0.9

Figure 25. GI Agent Personality Traits

The effects of the personality traits listed above are minimized in this experiment as to ensure the stability of the organizational structures; they could be used to explore other areas of the complex environment of infantry combat. Such areas are the role of leadership or training in combat.

Combat Parameters of the agents are likewise uniform among the agents with the exception of the sniper agents. Sniper agents have significantly better sensing range and weapons capabilities to give them approximately twice the combat power of the rifle soldier agents.

Combat parameters for rifle soldier agents and sniper agents are listed separately below.

Rifle Soldier Combat Parameters

Parameter	Value
Visual Sensing Range	6.00
Weapons Range	4.00
Probability to Hit	0.50
Movement Range	1.00
Durability	5.00

Figure 26. Rifle Soldier Combat Parameters

Sniper Combat Parameters

Parameter	Value
Visual Sensing Range	12.00
Weapons Range	8.00
Probability to Hit	0.90
Movement Range	1.00
Durability	5.00

Figure 27. Sniper Combat Parameters

C.   GI AGENT ORGANIZATIONAL EXPERIMENT RESULTS

Each of the three treatment organizations was tested against three different missions Attack, Defend and Movement to Contact. These three missions were chosen because they represent the bulk of the combat missions performed by an infantry company.

All treatment organizations were run 55 times against each of the three mission scenarios. Each treatment infantry has a total of one hundred three agent soldiers in the unit.

The results of the three experiments are detailed separately below.

1. Attack Mission

The attack mission produced a mixed bag of results. Depending on how the mission was measured resulted in a more favorable rating for either the squad level treatment or the company level treatment. The platoon level treatment produced significantly worse results than either of the other two treatments.

The measures of effectiveness for this mission were the average number of blue force killed-in-action and the average number of blue force agents that arrived and remained in the mission objective area. The killed-in-action statistic measures the survivability for the individual agent in the treatment organization. The number of agents in the mission objective area is a measure of the organization's structural integrity. In essence how well a unit holds together in a combat situation.

An attack mission was considered a success if 75 percent of the unit was still alive and 60 percent of the unit was consolidated on the objective at the end of a run. Seventy five percent of a unit's combat strength is considered to be the threshold at which a unit can still continue to function or be combat effective. This is an arbitrary measurement just meant to provide contrast between the treatment organizations.

The average for the squad level treatment was 18.18 with a standard deviation of 4.65 agents kill-in-action.

The average for the platoon level treatment was 38.72 with a standard deviation of 10.43 agents kill-in-action.

The average for the company level treatment was 24.38 with a standard deviation of 5.96 agents kill-in-action

These results are a strong argument for organizing a unit in the manner described by the squad level treatment.

The killed-in-action averages for the three treatments are shown below in a box plot that contains the standard deviation range around the average.
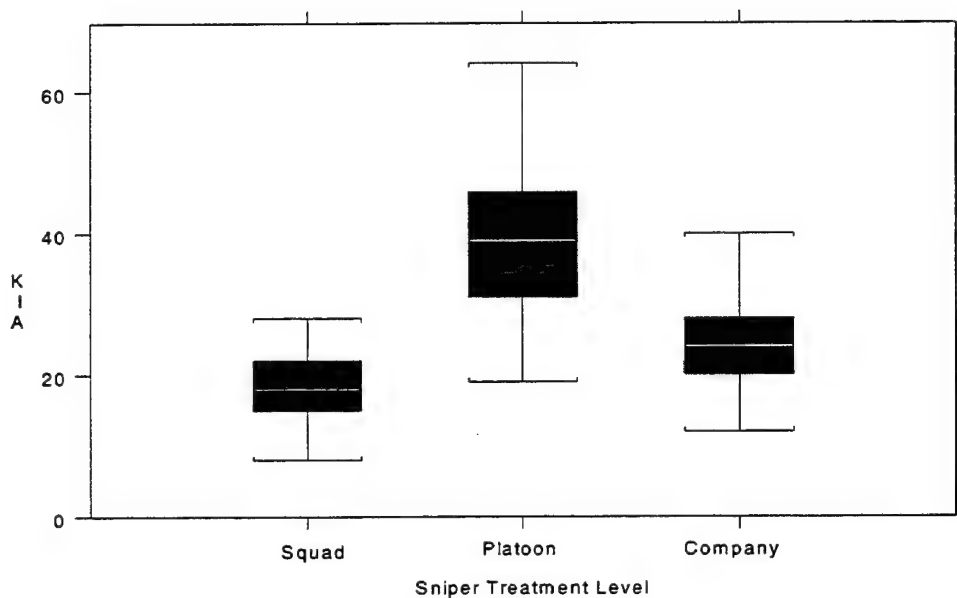
Figure 28. Average KIA for Attack Mission

The average KIA results correlate to the results obtained from looking at the number of agents in the mission objective area at the end of a run. The mission objective area in the case of the attack mission is the area of terrain that the attacking force is trying to control. This measurement of effectiveness is a good way to measure how well a unit

maintained its integrity during the attack. A treatment company that has a high average does not have many soldier agents that quit or stop fighting. A low average correlates to a unit that loses structural integrity.

The figure below shows these averages.
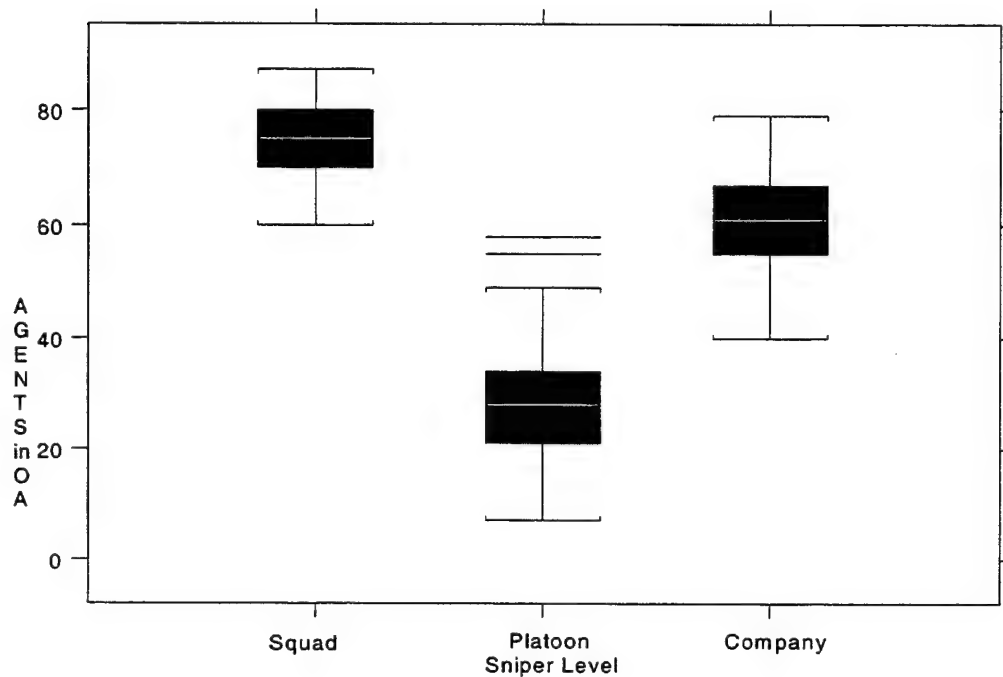
Figure 29. Average Agents in Attack Objective Area

The average for the squad level treatment was 74.87 with a standard deviation of 5.74 agents kill-in-action.

The average for the platoon level treatment was 28.69 with a standard deviation of 11.54 agents kill-in-action.

The average for the company level treatment was 61.4 with a standard deviation of 9.3 agents kill-in-action

These results confirm an advantage for the squad level treatment organization. The average run at the squad treatment level produced a mission success rate of 96.3 percent.

More surprising are the results concerning the platoon level treatment. In all fifty five runs the platoon level treatment organization failed to achieve a single mission success.

The mission success rating for the company level treatment organization was 47.2 percent.

The reasons for differences in the treatment organizations are fairly obvious once several runs are observed. With the squad level treatment the snipers are usually the first blue elements to engage the red forces. The superior sensor and weapons range of the snipers allows them to engage the forward positioned red forces without receiving any return fire. The snipers longer sensor range gives the blue force advanced warning and allows the blue leaders to concentrate there forces against the red units. These two factors are the crucial difference for the squad level treatment company.

For the platoon and company treatments the snipers are controlled by the platoon leaders and the company commander respectively. Because they are under the control of a higher-level leader the snipers are not the first soldiers into the fight. This negates the sensor range advantage that the snipers can provide the blue forces. Without the advanced warning from the lead snipers the blue forces tend to enter the red forces engagement range in a more piecemeal fashion. The advantage of superior firepower that snipers provide is negated in the platoon level treatment. The three platoon leaders each control a sniper section. Each platoon leader will make a different decision as to the time and place

55

that he will commit the sniper section. This results in the sniper sections being committed to the fight one at a time. Sniper employment in this fashion results in too little force employed at a critical time to affect the outcome of the battle.

For the company level treatment the snipers are generally committed to the fight later than in the platoon level treatment but they are committed in mass. This allows the snipers to completely dominate the area of the battlefield that they occupy.

2. Movement to Contact Mission

The results of this experiment were less conclusive than for the attack mission. After much analysis and some intuitive reasoning it was determined that the platoon level treatment provided the best results. Although, only slightly better than the other treatment organizations.

The same measures for effectiveness used in the attack experiment were unable to provide enough information, so different measures were used. The new measure of effectiveness is the force ratio between the blue and red forces after combat has ceased. This provided a measurement of how much damage the blue forces did as compared to how much damage was done to the blue forces.

The average ending force ratio for the squad level treatment was 1.09 to 1 with a standard deviation of 0.42.

The average ending force ratio for the platoon level treatment was 1.31 to 1 with a standard deviation of 0.55

The average ending force ratio for the company level treatment was 1.16 to 1 with a standard deviation of 0.56

The chart below shows the average force ratio by treatment level with standard deviation range.
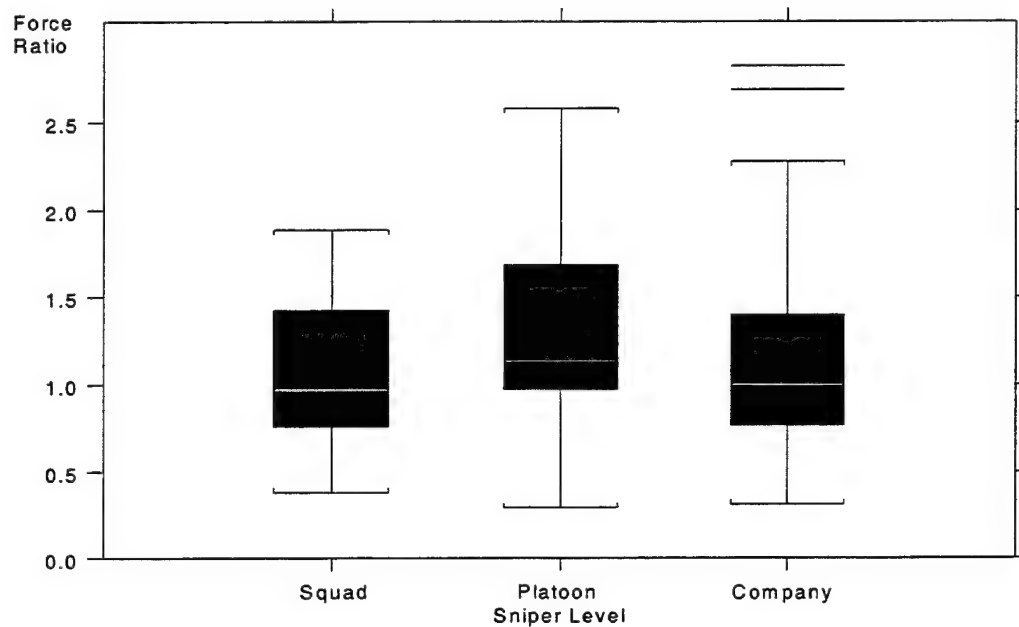
Figure 30. End of Run Blue vs Red Force Ratios

Looking at the averages would tend to indicate that there is not a significant difference between the treatments. However, a simple histogram of each treatment ending force ratio show some significant differences. The platoon level histogram showed a distinct bivariate curve as opposed to the more normal curves of the squad and company level treatments. This result of the platoon level treatment indicates that when organized in this manner the infantry company tend to at least break even or do very well in the scenario described in this experiment.

The histograms for each treatment are show below.
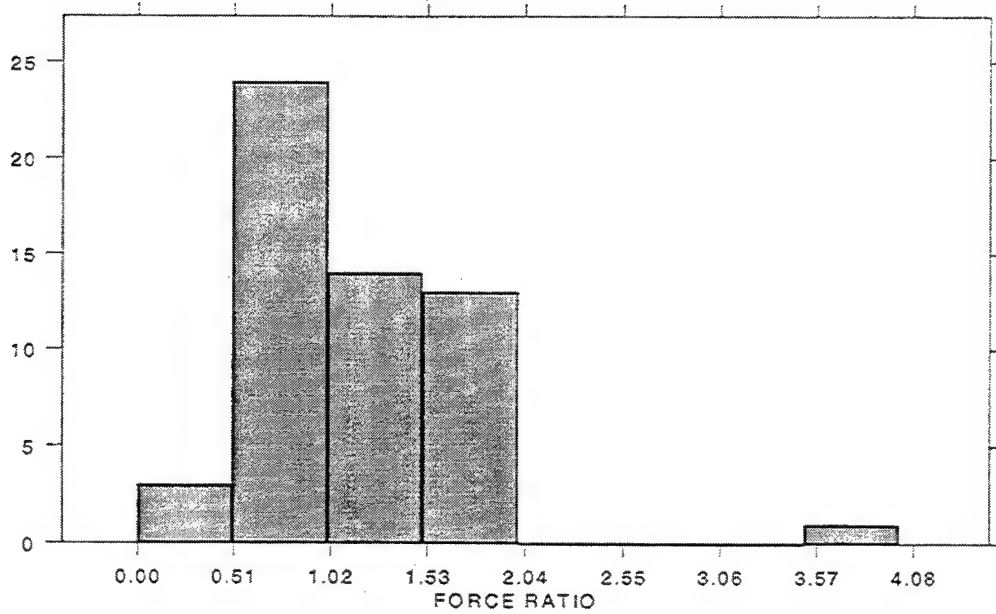
Figure 31. Squad Level Ending Force Ratios
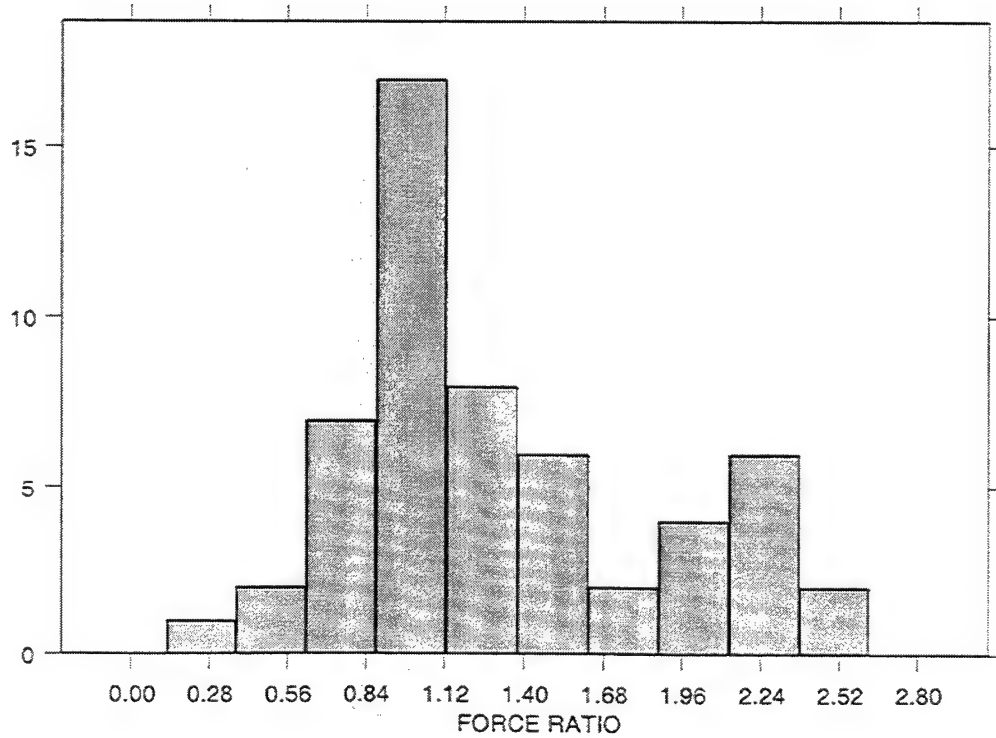
Figure 32 Platoon Level Ending Force Ratios
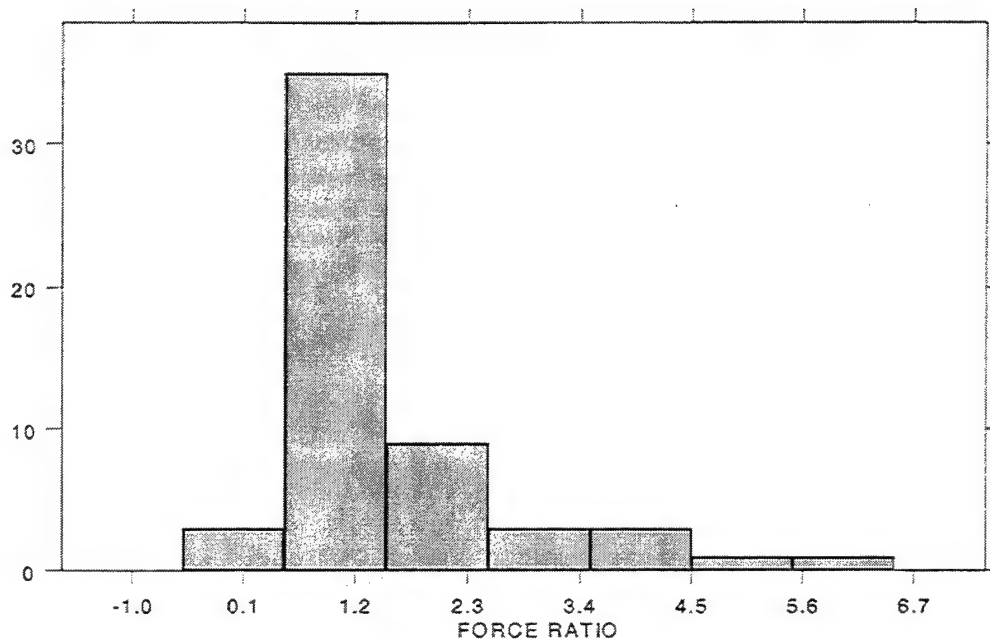
Figure 33. Company Level Ending Force Ratios

Examining the histograms shows that the majority of the runs for the platoon level

treatment resulted in a favorable force ratio at the end of the run. This result lends

credence to the theory that the movement to contact or meeting engagement fight is

primarily a platoon level fight. Instead of one big company versus company fight the

movement to contact fight is three smaller, platoon versus platoon fights. Having a three-

man sniper section reinforce each platoon provides a significant firepower advantage to

the individual platoon. With the companies dispersed more in the movement to contact a

three-man sniper section can provide a platoon with local dominance of the battlefield.

Enabling the platoon to win a one on one fight with another platoon.

The reasons that the squad and company level treatments performed to a lower

standard are different. For the squad level treatment single snipers could not do enough

damage to a force that was closing rapidly with them. This reduction in time and space,

coupled with the dispersed nature of the snipers deployment negated most of the advantages of the snipers. The oncoming enemy forces overwhelmed individual snipes before they could significantly influence the battle. The exception was when the squad level treatment did well versus the enemy force. In this case the squad snipers had migrated to the back of the squad formations. From the back of the squad formation, a sniper could engage the enemy from an over the shoulder position behind his squad mates. This protection significantly increased the lifespan of a sniper. However when squad level treatment with the snipers in the back of the squads was compared to the platoon level treatment, the two overall company formations looked very similar. This insight lends more recognition to the validity of the platoon level treatment as the most effective for the movement to contact mission.

The problems for the company level treatment were just the opposite. Faced with the problem of three platoons in need of reinforcement, the company commander had to choose one and let the other two platoons fend for themselves. If the two unreinforced platoons did well the company did well, if they did not then the company did not do well.

This resulted in wildly variable force ratios at the end of the runs for the company level treatment. However, the higher force ratios generally corresponded to extremely heavy casualties on both sides, often leaving less then a dozen blue agents surviving and only a few red agents surviving. Giving a high amount credit to the company level treatment when the unit was essentially destroyed does not make sense. The five highest force ratios were removed from the company treatment level for the calculation of the force ratio average. These results are represented in the company level histogram, but not in the force average box plot.

60

3. Defend Mission

The results of the defend experiment provided some unique insights into how an organization fights. Although all three organizations kept the enemy force from achieving mission success as defined in the attack mission. The most successful of the three treatments was the squad level treatment.

GI Agent does not having the capability of representing defensive obstacles and emplacements. Without this augmentation the defending unit was given essentially a "die in place" mission. The resulting casualties for the defending force were very high for all three treatments. This negated the usefulness of using KIA data to analyze the organization. The measure of effectiveness that was used was the number of surviving attacking agents in the attacking forces mission objective area at the end of a run. Although an indirect measurement, this provided an objective standard to analyze the strengths and weakness of the organizational structures of the three treatments.

The measure of effectiveness based on how well the enemy force did, may not seem to be the most accurate means of measuring the effectiveness of an organizational structure. But, the goal of this experiment is to gain insight into the dynamic of infantry combat.

The average number of enemy agents in the objective area for the squad level treatment was 44.15 with a standard deviation of 14.43.

The average number of enemy agents in the objective area for the platoon level treatment was 58.56 with a standard deviation of 9.09.

The average number of enemy agents in the objective area for the company level treatment was 65.4 with a standard deviation of 10.32.

61

The diagram below is a box plot of the three treatments with a standard deviation range.
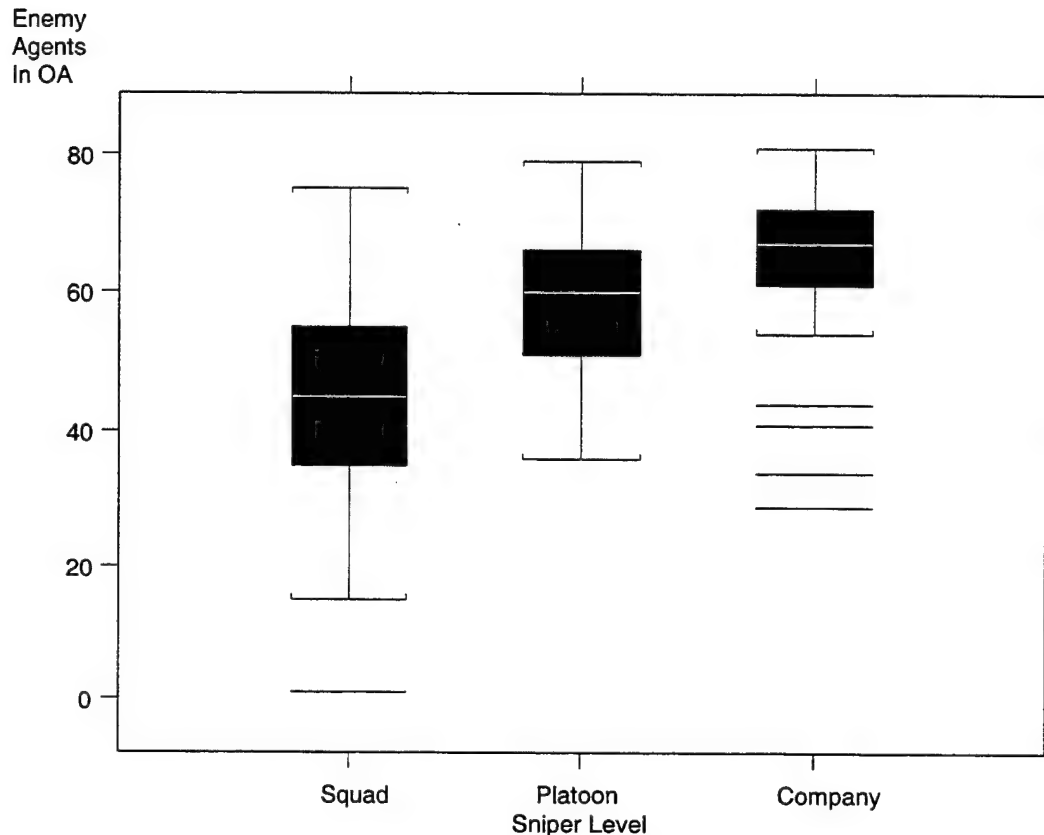
Enemy
Agents
In OA

Figure 34. Average Enemy Agents in Objective Area

The squad level treatment resulted in a lower number of enemy forces in the attack objective area because sniper elements were able to effectively operate on the flanks of the attacking force. The snipers in the squads of the flank platoons were in a position to engage the attacking force without being engaged in return. Without a direct threat to the unit they were the flank snipers were allowed to operate independently. Often several snipers would form an ad hoc unit and engage the attacking force from the flank. This is an indirect result of the squad level treatment. An example of this behavior is show in the figure below.
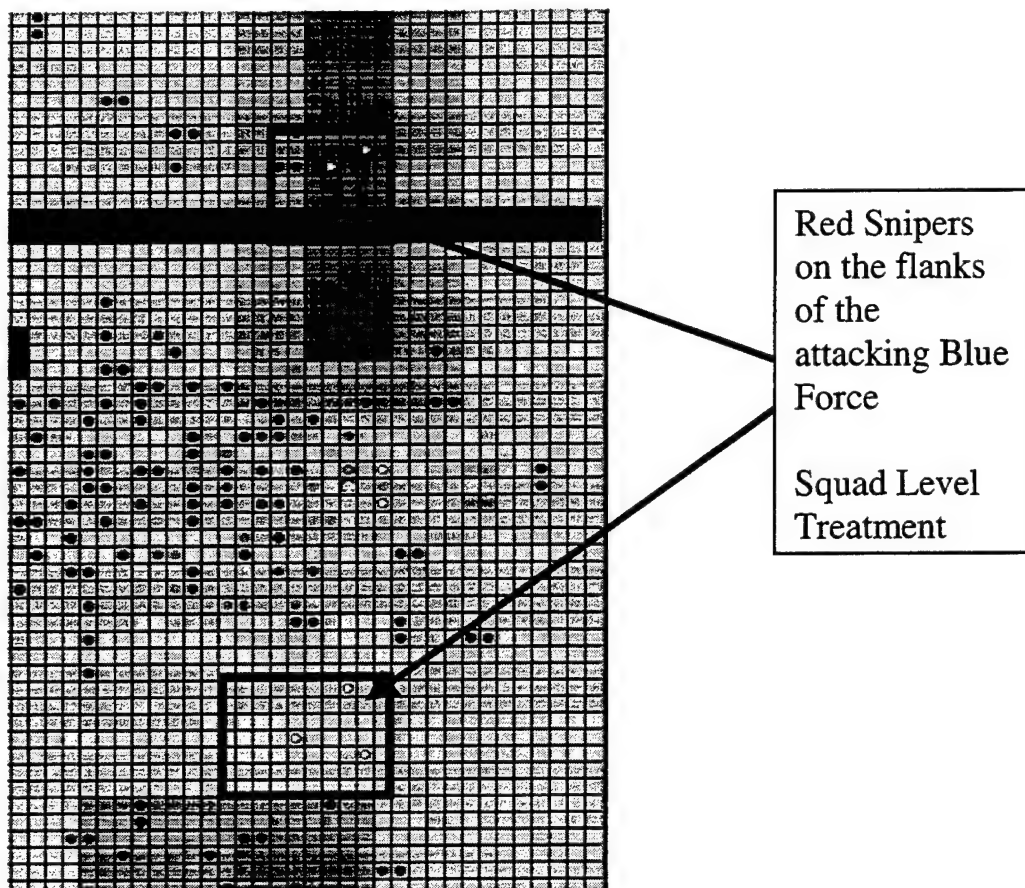
Figure 35. Red Snipers engage attacking Blue Force

A reason the platoon level treatment was less effective had to do with the fact that the attacking force concentrated on attacking the center platoon. The two platoons on flanks generally did not commit their snipers to the fight until they were threatened by the attacking force or were directed to do so by the company commander. This was generally to little to late to make a difference as unit integrity had already started to breakdown by this time.

In the case of the company level treatment, it was somewhat surprising that this treatment did the worst of the three. However, watching several runs indicated that the defensive company commander was unable to react a timely manner and commit the

63

snipers to the fight when it would have made a difference. This is probably the result of no advanced warning as to the impending attack. The snipers were generally committed to the fight after the center platoon was decimated and were at close range to the enemy force. The snipers were then simply overwhelmed by the superior numbers of the attacking force.

4. Final Discussion

The experiments presented in this chapter were intended to demonstrate the ability of the GI Agent MAS simulation to produce plausible results. GI Agent certainly possesses limitations that hinder its ability to represent all tactical aspects and employment techniques. But it should be kept in mind that GI Agent is an initial simulation tool. Future enhancements and modifications will only add to the usefulness and capabilities of GI Agent. Suggestions for future work, enhancements, and modifications are addressed in Chapter V.

The agents in this thesis do not completely represent real soldiers in any meaningful way. Nor are they intended to model the combat decision making processes used by infantrymen in combat. They are designed to simulate the actions of infantrymen. In this regard, they are capable of describing the actions of infantrymen through the goal satisfaction paradigm described in Chapter III.

Since the agents are not real infantrymen, the results of this experiment are not real. The results are however instructive in that they conform to the basic principles of combat. Mass and concentrated firepower provide the same advantage in GI Agent as they do in real combat. Organizational integrity is maintained by the chain of command

in GI Agent the same as in a real infantry unit. With this correlation in effects GI Agent

simulates real infantry combat in a way not done prior to its creation.

THIS PAGE INTENTIONALLY LEFT BLANK

V. CONCLUSIONS AND RECOMMENDATIONS

A. FUTURE WORK

GI Agent is a foundation for the development of a Multi-Agent System that simulates single entity level combat. GI Agent is designed to readily expand to include a multitude of soldier types and equipment. The artificial intelligence framework in GI Agent could readily be used in an analytical model or in a human-in-the-loop training simulation. In its current state, GI Agent is a limited analytical simulation.

1. Organizational Genetic Algorithm

The groundwork is laid in GI Agent to facilitate a genetic algorithm process for evolving a unit organizational structure. The idea being to input manpower and equipment available through an iterative genetic algorithm too evolve the organizational structure of a unit. This system would try to create the most effective organizational structure based on the resources provided.

2. GI Agent Soldiers that Learn

Build evolving GI Agents; incorporate a genetic algorithm into the simulation based on the goal/rule selection hierarchy in GI Agent. The simulation could agents to die and learn from their mistakes. The idea is to create a soldier agent that can improve its performance over time. Using a GA to enable soldier agents to learn could be a way to improve the organizational structure of a unit due to the increased effectiveness of the soldiers.

3. Expand the Sensor Array of the GI Agents

In the current version of GI Agent, the agents have only one way to detect hostile forces, the visual sensed environment. The Sensed Environment framework was

67

established to allow for expansion of the GI Agent's possible sensors. The paradigm could be extended to simulate thermal imaging, or ground surveillance radar for example.

4. Realistic Weapons Affects

Currently the GI Agents are armed with a generic rifle and a generic sniper rifle. The probability of hit numbers for these weapons are arbitrary. More accurate analysis can be done if the weapons used by the GI Agents are based on real data. Additional weapons types could also be added with relative ease.

5. Increase the Heterogeneity of the Units

GI Agent has two types of soldiers, riflemen and snipers. Numerous additional soldier skills could be added such as grenadier, machine gunner or engineer. This would greatly increase the fidelity and realism of the simulation. In addition unit vehicles could be added to the simulation. The possibilities are only limited by the imagination.

6. Cognitive Analysis of GI Agent Leaders

Installing a cognitive decision making model in the leaders of GI Agent could allow for analysis of the combat decision making process. GI Agent has a built in interface to allow an analyst to peer into the goal selection process of an agent. This could provide the foundation for a cognitive model interface and implementation. Analysts could study not just the how of a decision but the why of that decision.

B. LESSONS LEARNED

In designing a large software project like GI Agent, the software designer should consider researching the design of programs that manage large number of entities. Management of large numbers of entities is problematic in of itself. Development of the

68

software for this thesis would have benefited from a better understanding of the difficulties involved in handing a large number of entities.

Management of large numbers entities poses several problems. First in a time step simulation like GI Agent steps must be taken to reduce the computation time of a time step if the simulation is to run in a reasonable amount of time overall. Additionally, memory management becomes a big issue, as running a simulation like GI Agent is memory intensive.

C. CONCLUSION

The heart of GI Agent is the ability to simulate an infantry company allowing the organization of the unit to influence the outcome of combat. Although much more could be done to improve the fidelity of the simulation, the foundation for a multi-agent system capable of simulating entity level combat has been laid. GI Agent provides a unique way to model combat at the entity level by combining agent based programming techniques with advanced search algorithms and rule sets based on military doctrine.

One of the key problems for military professionals when viewing combat simulations is that the simulation does not "look right". GI Agent incorporates basic tenets of military operations into the agent goal rule structure. This allows the different agent organizations to maneuver in a fashion similar to a real infantry unit. Having an analysis tool that passes the intuitively correct test is key to gaining the acceptance of the military professional. GI Agent is not the end of the road but a serious and significant step along it.

THIS PAGE INTENTIONALLY LEFT BLANK

GLOSSARY

AgentBrainLid: Java class that is an interface with the agents in GI Agent. The interface allows the user to see the current state of the agent.

GIAgentSimEnv: Java class that contains the environment for the Multi-Agent Simulation GI Agent.

GIJoelAgent: Java class that is the agents in GI Agent.

Sensed Visual Environment: the local area of the total environment relative to the agent and within its visual sensor range

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX A

RELATIONSHIP-ROLE-GOAL-RULE STRUCTURE

Relationship – ArmyRelationship (BlueArmy & RedArmy)

    Role – RifleSoldierRole (RedRifleSoldier & BlueRifleSoldier)

        Goal – EnsureSurvival

            Rule – DisengageEnemyRule

            Rule – MaximizeCoverRule

            Rule – MaximizeMoveSpeedRule

            Rule – MoveToPositionRule

        Goal – EngageEnemy

            Rule – ShootAnyPerceivedEnemyRule

            Rule – ShootClosestPerceivedEnemyRule

            Rule – ReturnFireRule

            Rule – ShootEnemyLeaderRule

    Role – SniperSoldierRole (RedSniperSoldier & BlueSniperSoldier)

        Goal – EnsureSurvival

            Rule – DisengageEnemyRule

            Rule – MaximizeCoverRule

            Rule – MaximizeMoveSpeedRule

            Rule – MoveToPositionRule

        Goal – EngageEnemy

Rule – ShootAnyPerceivedEnemyRule

Rule – ShootClosestPerceivedEnemyRule

Rule – ReturnFireRule

Rule – ShootEnemyLeaderRule

Relationship – SquadRelationship (BlueSquad & RedSquad)

Role – SquadLeaderRole

Goal – Atttack

Rule – MaximizeCoverRule

Rule – MassSquadRule

Rule – MoveWithUnitRule

Rule – MoveToPositionRule

Rule – MoveToObjectiveRule

Goal – Defend

Rule – MaintainSeparationRule

Rule – MoveToDefensiveObjectiveRule

Rule – MaintainStandoffRule

Rule – MoveToClosestEnemyRule

Goal – Recon

Rule – MoveWithUnitRule

Rule – MoveToPositionRule

Rule – MoveToReconObjectiveRule

Rule – MoveInDirectionRule

74

Role – SquadMemberRole (RifleSoldiers)

    Goal – ProtectFlankSoldier

        Rule – MoveToClosestFriendlyRule

        Rule – ReturnFireRule

        Rule – ShootAnyPerceivedEnemyRule

        Rule – MoveToClosestEnemyRule

    Goal – KeepSquadLeaderInformedGoal

        Rule – FullReportRule

    Goal – Atttack

        Rule – MaximizeCoverRule

        Rule – MassSquadRule

        Rule – MoveWithUnitRule

        Rule – MoveToPositionRule

        Rule – MoveToObjectiveRule

    Goal – Defend

        Rule – MaintainSeparationRule

        Rule – MoveToDefensiveObjectiveRule

        Rule – MaintainStandoffRule

        Rule – MoveToClosestEnemyRule

Goal – Recon

    Rule – MoveWithUnitRule

    Rule – MoveToPositionRule

    Rule – MoveToReconObjectiveRule

    Rule – MoveInDirectionRule

Relationship – PlatoonRelationship (BluePlatoon & RedPlatoon)

  Role – PlatoonLeaderRole

    Goal – Attack

      Rule – MaximizeCoverRule

      Rule – MassSquadRule

      Rule – MoveWithUnitRule

      Rule – MoveToPositionRule

      Rule – MoveToObjectiveRule

    Goal – Defend

      Rule – MaintainSeparationRule

      Rule – MoveToDefensiveObjectiveRule

      Rule – MaintainStandoffRule

      Rule – MoveToClosestEnemyRule

    Goal – Recon

      Rule – MoveWithUnitRule

Rule – MoveToPositionRule

Rule – MoveToReconObjectiveRule

Rule – MoveInDirectionRule

Role – PlatoonMemberRole (Squad Leaders)

Goal – ProtectFlankSquad

Rule – MoveToClosestFriendlyRule

Rule – ReturnFireRule

Rule – ShootAllPerceivedEnemyRule

Rule – MoveToClosestEnemyRule

Goal – KeepPlatoonLeaderInformedGoal

Rule – FullReportRule

Relationship - CompanyRelationship

Role – CompanyCdrRole

Goal – Recon

Rule – DispersePlatoonsRule

Rule – MaximizeCoverRule

Rule – MaximizeMoveSpeedRule

Rule – MoveToPositionRule

Goal – Defend

Rule – MassPlatoonsRule

Rule – DispersePlatoonsRule

Rule – MaximizeCoverRule

Rule – MaximizeMoveSpeedRule

Rule – MoveToPositionRule

Goal – Attack

    Rule – MassPlatoonsRule

    Rule – DispersePlatoonsRule

    Rule – MaximizeCoverRule

    Rule – MoveWithUnitRule

    Rule – MoveToPositionRule

    Rule – MoveToObjectiveRule

Role – CompanyMbrRole (Platoon Leaders)

    Goal – ProtectFlankPlatoon

        Rule – MoveToClosestFriendlyRule

        Rule – ReturnFireRule

        Rule – ShootAnyPerceivedEnemyRule

        Rule – MoveToClosestEnemyRule

    Goal – KeepCompanyCommanderInformedGoal

        Rule – FullPlatoonReportRule

APPENDIX B

INSTALLING AND·RUNNING GI AGENT

The simulation provided in this thesis is an excellent source for unit organizational development using the GI Agent software. It is intended that future development start with the installation and running of this simulation The following instructions are intended to give the user specific instructions to install and run these Java-based applications.

1. Check to see if you have the latest Java build on your machine.
    a. At the "C prompt" type: **java –version**. You should see something like:
1. **java version "1.3.0"**

    Java(TM) 2 Runtime Environment, Standard Edition (build 1.3.0-C)
    Java HotSpot(TM) Client VM (build 1.3.0-C, mixed mode)
    The version should be "1.2.0" or higher.

2. If the latest Java JDK is not installed on the computer being used:
    a. Copy the "j2sdk1_3_0-win.exe" file to the computers desktop, or other temporary directory.
    b. Double click the icon to start installation, or run the ".exe" file. Java version 1.3.0 will be installed and set up on your machine.
    c. Java Docs 1.3 are also included on the CD if you're a developer and want the latest from documentation from Sun.

3. Copy the folder: "GIAgent" into a new folder of your own choice (I recommend a new folder called GIAgent).

4. To run the GI Agent simulation:

    a. Open a DOS window and move to the directory containing GI Agent.
    b. At the command prompt type "java GIAgentSim".

For the simulation developer: The complete code listing is included in the attached CD or is available at http://www.npsnet.org/~moves/GIAgent. I encourage any interested parties to look through the code and if there are any questions or comments, please contact use through

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

Amit P., *The A* Algorithm*,
http://theory.stanford.edu/~amitp/GameProgramming/AstarComparison.html, 3
October, 2000.

Atkin, Marc S., Westbrook David L. and Cohen Paul R., *"Capture the Flag: Military
Simulation Meets Computer Games"*. AAAI Spring Symposium on AI and Computer
Games, 1999.

Atkin, Marc S., King, Gary W., Westbrook, David L., Heeringa, B., Hannon, A., and
Cohen, P., *"Hierarchical Agent Control: A Framework For Defining
Agent Behavior"*, 2001.

Ferber, J., *Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence*,
London: Addison-Wesley, 1999.

Headquarters, Department of the Army, *Cavalry Operations* (Field Manual 17-95),
Washington, DC: U.S. Government Printing Office, 1996.

Headquarters, Department of the Army, *Cavalry Troop* (Field Manual 17-97),
Washington, DC: U.S. Government Printing Office, 1995.

Headquarters, Department of the Army, *Infantry-Based Opposing Force* (Field Manual
100-63), Washington, DC: U.S. Government Printing Office, 1996.

Hiles, J. (1999). *Course Notes for MV-4015 Agent-Based Autonomous Behavior for
Simulations.* Winter, 2000, Naval Postgraduate School.

Ilachinski, A., Center for Naval Analyses, *ISAAC Irreducible Semi-Autonomous Adaptive
Combat: An Artificial-Life Approach to Land Warfare*, http://www.cna.org/isaac/, 24
July 00.

ISAAC (2000). *Irreducible Semi-Autonomous Adaptive Combat (ISAAC): An Artificial-
Life Approach to Land Warfare.* http://www.cna.org/isaac/extabs.htm, 30 July 00.

Ilachinski, A., *Irreducible Semi-Autonomous Adaptive Combat (ISAAC): An Artificial-
Life Approach to Land Warfare (U)*, (Center for Naval Analyses Research Memorandum
CRM 97-61.10), Alexandria, VA: Center for Naval Analyses, 1997.

Kawick, M., *Real-Time Strategy Game Programming*, Plano, TX Woodware Publishing,
1999.

Reynolds, C. W. (1987) "Flocks, Herds, and Schools: A Distributed Behavioral Model", in *Computer Graphics*, 21(4) (SIGGRAPH 87 Conference Proceedings) pages 25-34. http://www.red3d.com/cwr/papers/1987/boids.html, 30 July 00.

Roddy K. & Dixon M., Master's thesis, Naval Postgraduate School, Monterey, CA, 2000.

Stine, J., *Representing Tactical Land Navigation Expertise*, Master's thesis, Naval Postgraduate School, Monterey, CA, 2000.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center 2
   8725 John J. Kingman Road, STE 0944
   Ft. Belvoir, Virginia 22060-6218

2. Dudley Knox Library 2
   Naval Postgraduate School
   411 Dyer Road
   Monterey, California 93943-5101

3. Dr. Donald Gaver 1
   Operations Research Dept.
   Naval Postgraduate School
   Monterey, California 93943-5221

4. Dr. Michael J. Zyda, Code CS/Zk.. 1
   Chair, Modeling, Virtual Environments and Simulation (MOVES)
   Naval Postgraduate School
   Monterey, CA 93943-5101

5. Mr. John Hiles 1
   MOVES Academic Group
   Naval Postgraduate School
   Monterey, California 93943-5118

6. CPT Joel S Pawloski 1
   211 Salerno St.
   Seaside, CA 93955

7. Dr. Alfred Brandstein 1
   MCCDC (C 45)
   Studies and Analysis Division
   3300 Russell Road
   Quantico, Virginia 22134-5001

8. Dr. Rudolph P. Darken 1
   MOVES Academic Associate
   Code CS/DR
   Naval Postgraduate School
   Monterey, Ca 93943

9. Curtis Blais..1
    Institute for Joint Warfare Analysis
    Naval Postgraduate School
    Monterey, CA 93943-5101

10. David Laflam...1
    7033 Strathmore St. Apt#4
    Chevy Chase MD 20815

11. Dale Henderson..1
    11766 Gascony Pl
    Woodbridge, VA 22192

12. Leo Pawloski..1
    3733 Stoney Creek Rd
    Oakland, MI 48363